Hochschule Offenburg
offenburg.university

Fakultät Elektrotechnik, Medizintechnik und Informatik

Bachelorthesis

zum Erlangen des akademischen Grades Bachelor of Engineering (B.Eng.)
im Studiengang

Elektro- und Informationstechnik

Entwicklung eines energieautarken Türschildes mit NFC-Konfigurationsschnittstelle, E-Paper-Display und zugehöriger Android-App

von

Andreas Angermayr

betreut durch

Prof. Dr.-Ing. Elke Mackensen

und

Prof. Dr.-Ing. Daniel Fischer

im Bearbeitungszeitraum

1. September 2019 - 29. Februar 2020

Eidesstattliche Erklärung

Hiermit versichere ich eidesstattlich, dass die vorliegende Arbeit mit dem Titel

Entwicklung eines energieautarken Türschildes mit NFC-Konfigurationsschnittstelle, E-Paper-Display und zugehöriger Android-App

von mir selbstständig und ohne unerlaubte fremde Hilfe angefertigt worden ist, insbesondere, dass ich alle Stellen, die wörtlich, annähernd wörtlich oder dem Gedanken nach aus Veröffentlichungen, unveröffentlichten Unterlagen und Gesprächen entnommen worden sind, als solche an den entsprechenden Stellen innerhalb der Arbeit durch Zitate kenntlich gemacht habe, wobei in den Zitaten jeweils der Umfang der entnommenen Originalzitate kenntlich gemacht wurde.

Die Arbeit lag in gleicher oder ähnlicher Fassung noch keiner Prüfungsbehörde vor und wurde bisher nicht veröffentlicht. Ich bin mir bewusst, dass eine falsche Versicherung rechtliche Folgen haben wird.

Biberach, am 27. Februar 2020

Andreas Angermayr

Kurzfassung

In dieser Bachelorthesis wurde ein Fuktionsmuster eines energieautarken elektronischen Türschildes mit einem 7,8” großen E-Paper-Display und NFC-Konfigurationsschnittstelle entwickelt, auf dem per Smartphone-App und NFC einfach Informationen wie Abwesendheitsnachrichten angezeigt werden können. Hierzu wird ein Kommunikationsprotokoll entwickelt, welches die Kommunikation zwischen App und Türschild spezifiziert, und einen Befehlssatz zur Konfiguration des Türschildes bereitstellt. Das System wird aus amorphen Silizium-Solarzellen versorgt und verfügt über einen LiPo-Akku als Energiespeicher. Durch sorgfältiges Hardware- und Softwareseitiges Low-Power-Design beträgt die Leistungsaufnahme im Ruhemodus lediglich $1,5\,\mu W$. Bedingt durch den anwenderfreundlichen, jedoch für Low-Power-Designs ungeeigneten Display-Controller, beträgt der Energieverbrauch während eines Updates $300\,mW$. Trotzdem zeigt sich, dass das System bei einer Zellfläche von knapp $220\,cm^2$ auch bei schlechter Beleuchtung von $10\,lx$ in dunklen Gängen mehrere Türschild-Updates pro Tag bereitstellen kann.

Abstract

Within the framework of this bachelor's thesis, a functional model of an energy-self-sufficient electronic roomplate with an 7.8" sized e-paper display and NFC configuration interface was developed, on which informations such as away-messages can be displayed using NFC and a smartphone app. For this purpose a communication-protocoll has been developed, which specifies the communication between app and roomplate and provides a set of commands for the configuration of the roomplate. The system is powered by amorphous silicon solar cells and has a LiPo accumulator for energy storage. By carefully selected low power design in hard- and software, the power consumption is only 1.5 µW during idle-state. Caused by the user-friendly, but for low power designs unsuitable display-controller, the power consumption during an update is 300 mW. Nevertheless it becomes apparent, that the system with an cell area of allmost 220 cm^2, even under a bad illuminance of 10 lx in dark corridors, can provide multible roomplate updates per day.

Danksagungen

Dieser Abschnitt ist den Menschen gewidmet, die direkt oder indirekt zum Erfolg dieser Arbeit beigetragen haben.

Guter Rat ist bekanntlich schwer zu finden, daher möchte ich mich zuerst bei Frau Prof. Dr.-Ing. Elke Mackensen bedanken, für stets hilfreiches Feedback und die Möglichkeit, während des Studiums über den Tellerrand der regulären Inhalte hinauszuschauen.

Ferner bedanke ich mich bei den Professoren der Hochschule Offenburg für erstklassige Lehre, namentlich Prof. Dr.-Ing. Daniel Fischer, Prof. Dr.-Ing. Werner Reich, Prof. Dipl.-Ing. Peter Gröllmann und Prof. Dr.-Ing. Stephan Pfletschinger.

Durch hervorragenden Code wesentlich zu dieser Arbeit beigetragen hat Herr Arne Diehm, der in der Skeid-Gruppe die Firmware des ersten Funktionsmusters entwickelte. Für spannende Diskussionen, Tipps und Hinweise von unschätzbarem Wert für diese Arbeit bedanke ich mich bei Herrn Ing. Patrick Moser.

Außerdem bedanke ich mich bei Herrn Harald Angermayr, dafür dass er mich lehrte Hände und Kopf zu benutzen, sowie für die Unterstützung bei der Anfertigung des Funktionsmusters dieser Arbeit. Frau Christa Büdel danke ich für einen nie endenden Support durch Kuchen. Frau Vivien Kerkoč danke ich besonders für hervorragende Verpflegung und Unterstützung in jeder Hinsicht.

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

EInk E-Ink-Display, Synonym für EPD

EPD E-Paper-Display

IDE Integrated-Development-Environment

LDO Low-Drop-Out-Regulator

MCU Micro-Controller-Unit

MPP Maximum-Power-Point

MPPT Maximum-Power-Point-Tracking

SDK Source-Development-Kit

SKEID Smartphone-Konfigurierbares-E-Ink-Display

SMU Source-Measure-Unit

Kapitel 1

Problemstellung

1.1 Motivation dieser Arbeit

Vor den Türen fast aller Büros und Ämter sind Türschilder angebracht, welche die Raumnummer, den Namen der dort Arbeitenden und gegebenenfalls die Öffnungszeiten enthalten. Zur Urlaubs- oder Mittagszeit finden sich dann oft Notizzettel, mit Klebeband an der Tür befestigt, auf denen Abwesenheitsinformationen hinterlassen werden.

Hierbei finden auch elektronische Türschilder zunehmend Anwendung, zumeist über eine W-Lan-Schnittstelle konfigurierbar und batteriebetrieben, oder über die Netzspannung versorgt. Bedenkt man die Anzahl der Türschilder in öffentlichen Gebäuden wie der Hochschule Offenburg, kann hierbei ein signifikanter Energiebedarf entstehen. Dieser ist im Zuge allgemeiner Bemühungen zur Reduktion des Energieverbrauchs kaum zu rechtfertigen, besonders weil die o.g. *klassische* Lösung mittels Notizzettel und bedrucktem Türschild keinerlei Energie benötigt.

Da die Forschungsgruppe um Prof. Dr.-Ing. Elke Mackensen[1] an der Hochschule Offenburg intensiv an Themen wie Ultra-Low-Power, energieautarke elektronische Systeme, Energy-Harvesting und NFC forscht, entstand die Idee, ein vollständig energieautarkes Türschild, versorgt aus einer Solarzelle und mit E-Paper-Display (EPD), als besonders energiesparende Anzeige, zu entwickeln.

Weiterhin soll sich das Türschild komfortabel mittels einer Smartphone-App über eine NFC-Schnittstelle konfigurieren lassen. Innovativ ist dieser Ansatz, weil es der-

[1] Professur für Digitale Schaltungstechnik und mikroelektronischer Systementwurf. Fakultät Elektrotechnik, Medizintechnik und Informatik. Anschrift: Raum B116, Badstraße 24 in 77652 Offenburg. E-Mail: elke.mackensen@hs-offenburg.de

zeit (Stand: Januar 2020) keine kommerziell erhältlichen Türschilder gibt, die über eine NFC-Schnittstelle verfügen. Neben den o.g. statischen Informationen, sollen sich dann komfortabel und schnell Informationen wie z.B. Abwesenheitsnotizen auf dem Türschild darstellen lassen. Abb. 1.1.1 zeigt den grundsätzlichen Systemaufbau.

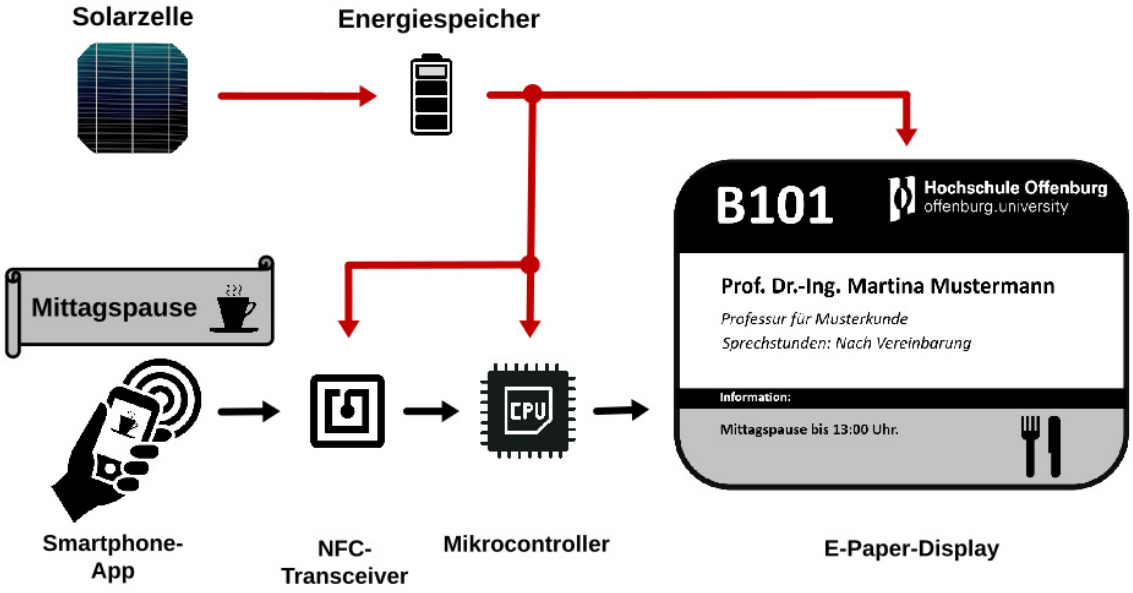


Abbildung 1.1.1: Grundsätzlicher Systemaufbau.

Im Wintersemester 2018 wurde durch eine sechsköpfige Projektgruppe (im Folgenden: *Die SKEID²-Gruppe*) im Rahmen des Labors *Systementwicklung* im Studiengang Elektro- und Informationstechnik an der Hochschule Offenburg ein Funktionsmuster eines solchen Türschilds entwickelt und aufgebaut, sodass die grundsätzliche Funktion des Systems verifiziert werden konnte[12]. Dieses Funktionsmuster ist, wie in Abb. 1.1.2 zu sehen, aus Development-Boards aufgebaut und wird über USB versorgt und ist damit nicht energieautark.

Ziel der vorliegenden Bachelorarbeit ist es nun, die von der Skeid-Gruppe verwendeten Komponenten zu reevaluieren, ein Energieversorgungskonzept und die zugehörige Hardware für das Türschild zu entwickeln und ein zweites Funktionsmuster aufzubauen. Anschließend muss die Skeid-Software portiert und zusammen mit der Smartphone-App (im Folgenden *App*) gegebenenfalls angepasst werden. Dann soll das neue Funktionsmuster hinsichtlich Energieverbrauch, Low-Power-Design und Benutzerfreundlichkeit analysiert und bewertet werden.

[2]Smartphone-Konfigurierbares-E-Ink-Display (SKEID)

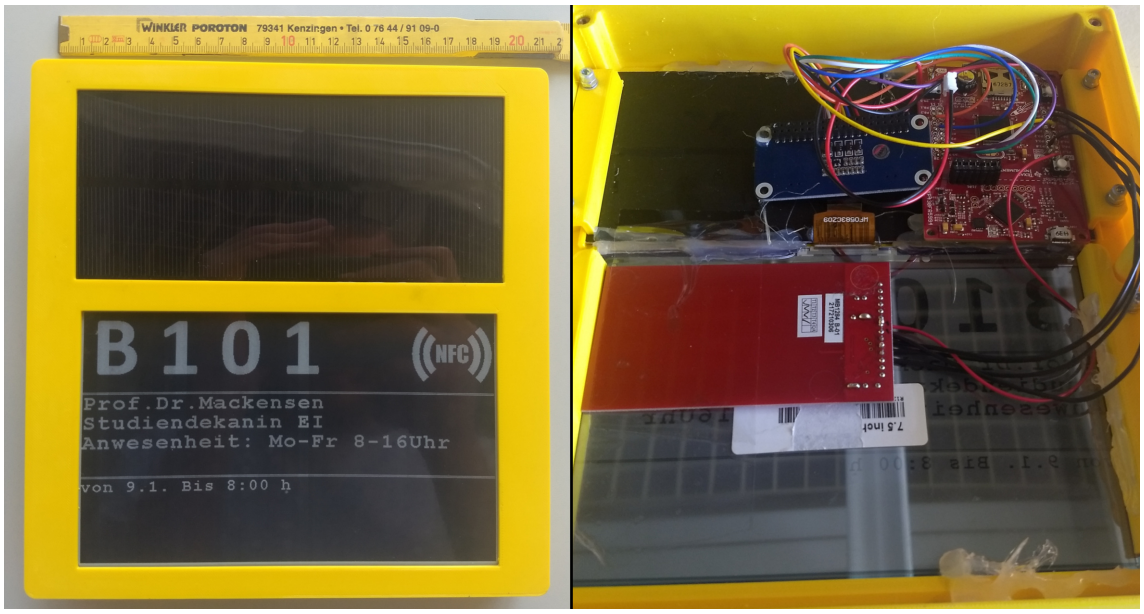


Abbildung 1.1.2: Funktionsmuster der Skeid-Gruppe

1.2 Aufbau der Arbeit

Die vorliegende Arbeit richtet sich an drei Lesergruppen und verfolgt damit verschiedene Ziele

1. Als Bachelorthesis muss sie die Tätigkeiten wärend der Bearbeitung der Bachelorarbeit für die Prüferin bzw. Zweitprüfer dokumentieren, aufzeigen dass die Zeit sinnvoll genutzt wurde und dass die angewandte Methodik der eines Ingenieurs entspricht.

2. Für eventuellen Folgearbeiten, die das entwickelte System erweitern, testen oder reevaluiluieren sollen, müssen die technischen Details und Designentscheidungen der Implementierung hinreichend tief dokumentiert werden, sodass z.B. Messungen oder Änderungen am Evaluationsboard oder der Firmware vorgenommen werden können.

3. Lesern die an Teilaspekten der Arbeit interessiert sind, z.B. der Auswahl der Solarzelle, sollen sich schnell und leicht einen Überblick über die Zusammenhänge verschaffen können, um die relevanten Abschnitte zu identifizieren und zu finden.

Grundsätzlich spiegelt die Gliederung dieser Arbeit die Arbeitsschritte die bei der Bearbeitung der Bachelorarbeit angefallen sind in chronologischer Reihenfolge.

Zunächst werden im Folgenden die Systemanforderungen spezifiziert. In Kapitel 2 wird der Stand der Technik vorgestellt, mit einer Marktübersicht bestehender Sys-

teme und den Schlüsseltechnologien dieser Arbeit wie E-Paper-Displays, Solarzellen und Energiespeicher.

Auf Basis dieser Darstellungen werden im Kapitel 3 die Systemkomponenten ausgewählt und gleichzeitig wichtige und grundlegende Designentscheidungen getroffen. Dies ist notwendig, weil hochoptimierte Systeme i.d.R eine starke Verzahnung der Komponenten erfordern, sodass die Auswahl einer Komponente die Auswahl anderer Komponenten beeinflusst. Dies mündet schließlich in ein Systemkonzept.

Im Kapitel 4 wird zunächst die genaue Verschaltung der Komponenten erläutert und auf kritische Punkte bzw. Probleme hingewiesen. Anschließend werden der Aufbau und die Funktionsweise der Mikrocontroller-Software (Firmware) erläutert, danach folgt ein Überblick zum Aufbau und Funktionsweise der Android-App.

Im Kapitel 5 schließlch werden die Mess- und Testergebnisse des Funktionsmusters vorgestellt und erläutert.

Basierend auf den Messergebnissen, wird in Kapitel 6 geklärt, inwieweit das System die gegebenen Anforderungen erfüllt. Hieraus ergeben sich Optimierungsansätze, die ebenfalls erläutert werden. Abschließend folgt eine Zusammenfassung und ein Ausblick.

1.3 Systemanforderungen

Die Systemanforderungen werden wie folgt festgelegt:

1. **Energieversorgung:**

   (a) **Das Türschild muss energieautark arbeiten.**
   D.h. die für den laufenden Betrieb notwendige Energie muss durch *Energy-Harvesting* aus der Umgebung gewonnen werden. Dies ist neben der komfortablen Konfigurierbarkeit das hauptsächliche Alleinstellungsmerkmal des Türschildes

   (b) **Das Türschild soll auf allen Systemebenen möglichst energie-sparend arbeiten.**
   Dies ist die notwendige Grundlage für die Energieautarkie des Türschildes, da nutzbare Umgebungsenergie in Gebäuden nur sehr begrenzt verfügbar ist.

   (c) **Dass System soll möglichst viele Updates pro Tag bereitstellen können, mindestens jedoch ein Update.**
   Dies beeinflusst nicht nur die Größe der Energiespeicher, sondern insbe-

sondere auch die Menge der Energie, die mindestens aus der Umgebung gewonnen werden muss.

(d) **Das Türschild soll möglichst unabhängig von den spezifischen Bedingungen des Aufhängungsortes sein.**
Das Türschild soll Anforderung 1c an möglichst allen Türen mit üblichen Bedingungen erfüllen, statt z.B. auf Türen mit einer außergewöhnlich hohen Beleuchtungsstärke beschränkt zu sein.

2. **Display:**

(a) **Als Displaytechnologie soll ein E-Paper-Display verwendet werden.**
Diese benötigt im Gegensatz zu LCD- oder OLED-Displays im statischen Betrieb keinerlei Energie und sind für den Eisatzzweck daher optimal geeignet.

(b) **Die Informationen müssen auf dem Display ausreichend groß dargestellt werden.**
Richtwert ist die Größe klassischer Türschilder.

(c) **Auf dem Türschild müssen komfortabel Abwesenheitsnachrichten hinterlassen werden können.**
Weiterhin sind die üblichen Informationen wie Raumnummer, Name, Öffnungszeiten darzustellen.

3. **Benutzer-Schnittstelle:**

(a) **Die Konfiguration des Türschilds und das Hinterlassen von Abwesenheitsnachrichten muss über eine Smartphone Schnittstelle erfolgen.**
Dies entlastet das Türschild selbst, da die eigentliche Benutzerinteraktion auf dem Smartphone getätigt wird und unterstützt so Anforderung 1b.

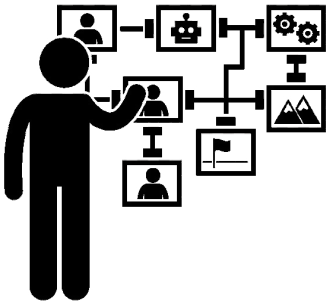
(b) **Als physische Schicht der Benutzerschnittstelle soll NFC verwendet werden.**
Zusammen mit 3a definiert dies den Ablauf der Benutzerinteraktion: Zunächst bereitet der Benutzer die gewünschte Aktion in der App vor, z.B. das Hinterlassen einer Abwesenheitsnachricht, abschließend hält er das Smartphone gegen das Türschild, um die Informationen, wie z.B. beim kontaktlosen Bezahlen zu übertragen. Die Konfigurierbarkeit über die NFC-Schnittstelle mit dem Smartphone ist neben der Energieautarkie das hauptsächliche Alleinstellungsmerkmal des Türschildes.

(c) **Die Schnittstelle soll sicher sein.**
D.h. Unbefugte sollen auch bei Kenntnis aller technischen Details des Systems nicht in der Lage sein, Konfiguration und Abwesenheitsnachrichten zu verändern.

4. **Smartphone-App:**

(a) **Die Smartphone-App muss auf aktuellen Android-Systemen lauffähig sein.**
Android hat als Smartphone-Betriebssystem bei weitem den größten Marktanteil.

Nachdem die Systemspezifikation nun festgelegt ist, wird im folgenden Kapitel der derzeitige Stand der Technik untersucht und dabei ein Systemkonzept erarbeitet.



Kapitel 2

Stand der Technik und Systemkonzeption

In diesem Kapitel wird der derzeitige Stand der Technik untersucht und darauf aufbauend ein grundlegendes Systemkonzept erarbeitet. D.h. es werden grundlegende Designentscheidungen festgelegt, sodass im nächsten Kapitel die Auswahl der Komponenten erfolgen kann. Zunächst werden daher vergleichbare Türschilder untersucht.

2.1 Marktübersicht vergleichbarer Türschilder

Wie eingangs erwähnt finden elektronische Türschilder immer weiter Verbreitung, entsprechend nimmt die Anzahl kommerziell erhältlicher Türschilder immer weiter zu. Tab. 2.1 listet einige derzeit erhältliche Türschilder auf und stellt sie anhand der Kriterien *Konfigurationsschnittstelle*, *Energieautark*, *Energiequelle*, *Displaytechnologie* einander vergleichend gegenüber.

Tabelle 2.1: Einige kommerziell erhältliche elektronische Türschilder.

Hersteller / Produkt	**Energie-quelle**	**Energie-autark**	**Konfigurations-schnittstelle**	**Display-technologie**	**Preis in €**
go2 / Türschild e-Ink	Batterie	-	USB	EPD	295
m.i.b. / door_ink	Batterie	-	"Funk"(W-Lan?)	EPD	?
e-shelf-labels / ROOMZ	Batterie	-	W-Lan	EPD	?

In Abb. 2.1.1 sind die Produkte dargestellt, die Tabellendaten stammen aus den Bildquellen.

Es wird klar, dass zwar kommerzielle Lösungen existieren, jedoch keine die energieautark sind und auch keine die über eine NFC-Konfigurationsschnittstelle verfügen. Die Konfiguration über USB ist wenig komfortabel, bei den W-Lan-

Abbildung 2.1.1: Marktübersicht vergleichbarer Türschilder [5], [6], [7]

Schnittstellen ist zwar eine Konfiguration mittels Smartphone denkbar, erfordert jedoch ein Einloggen in das W-Lan-Netzwerk. Damit bleibt diese Arbeit durch den bestehenden Markt weiter motiviert.

2.2 E-Paper Displaytechnologie

2.2.1 Grundlagen E-Paper

E-Paper-Displays sind Displays deren Betrachtungseindruck jenem von bedrucktem bzw. beschriebenem Papier nachempfunden ist, d.h. kennzeichnend sind:

- weiter Betrachtungswinkel
- gut lesbar auch unter Sonnenlicht
- reflektives Display ohne Hintergrundbeleuchtung
- guter Kontrast

Weiterhin kennzeichnend sind ein nahezu leistungsloser statischer Betrieb sowie eine sehr dünne (teilweise biegbare) Bauform. Denkbare Anwendungsbereiche waren und sind vielseitig und reichen mittlerweile von E-Book-Readern bis zu Benzinpreisanzeigen an Tankstellen. Erste Forschungsprojekte wurden daher bereits in den 80ern durchgeführt, in den frühen 2000ern waren erste Serienmodelle der Firma E-Ink auf dem Markt erhältlich. Der Firmenname *E-Ink* wurde dadurch zum Synonym für E-Paper-Displays.

2.2.2 Vergleich mit anderen Displaytechnologien

Zur Begründung der Verwendung eines EPD sind in Tab. 3.1 die drei Displaytechnologien LCD, OLED und E-Ink mit ihren jeweiligen Vor- und Nachteilen einander

Tabelle 2.2: Qualitativer Vergleich von LCD-, OLED- und E-Ink-Displays.

Kriterium	**LCD**	**OLED**	**EPD**
Auflösung (dpi)	++	+	++
Bildwiederholrate	+	++	–
Lesbarkeit (Sonne, Blickwinkel)	-	+	++
Energiebedarf	–	-	++

qualitativ vergleichend gegenübergestellt. Für diese Arbeit besonders ausschlaggebend ist der Energiebedarf. Wie oben angedeutet, benötigen EPDs lediglich für die Veränderung der Anzeige Energie, nicht aber im statischen Betrieb. Die Stromaufnahme von LCD- und OLED-Displays liegen üblicherweise in der Größenordnung Milliampere, womit das zu implementierende System nur mit EPDs zu realisieren ist.

2.2.3 Funktionsweise eines EPD

EPDs basieren auf dem Effekt der Elektrophorese, d.h. der Wanderung geladener Teilchen durch ein elektrisches Feld. In Abb. 2.2.1 ist der Querschnitt durch ein EPD-Panel dargestellt.

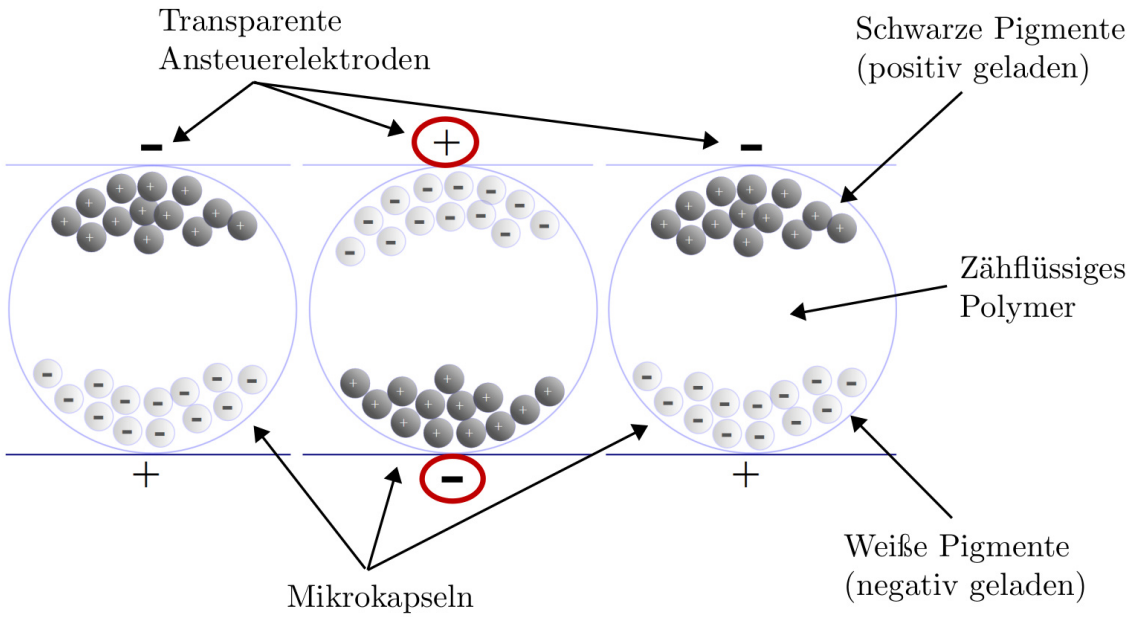


Abbildung 2.2.1: Querschnitt durch ein EPD-Panel.

Dieses besteht aus transparenten, sogenannten Mikrokapseln, die mit einem zähflüssigen Polymer und schwarzen und weißen Pigmenten gefüllt sind, die entgegengesetzt geladen sind. Auf bzw. unter dem Panel sind transparente Ansteuerelektroden angebracht, welche matrixartig angeordnet sind und letztlich die Pixel

definieren. Abb. 2.2.2 zeigt eine Mikroskopaufnahme eines Panels. Die Mikrokapseln sind gut zu erkennen, und es ist zu sehen dass ein Pixel aus mehreren unregelmäßig geformten Kapseln besteht.

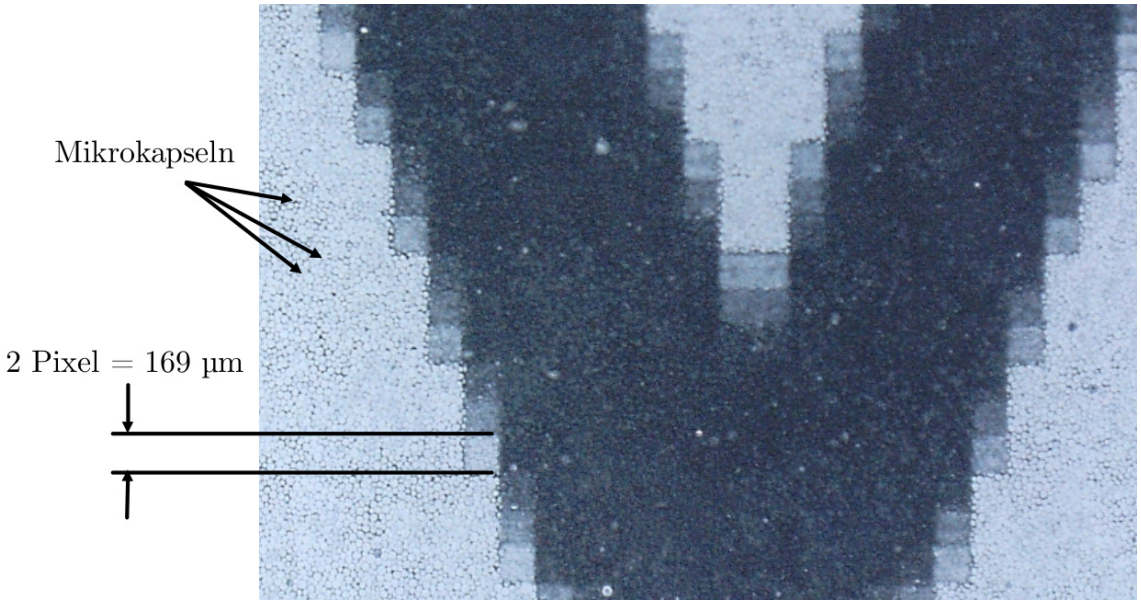


Abbildung 2.2.2: Mikroskopaufnahme eines EPD-Panels.

Durch das zähflüssige Polymer ist die Diffusion der Pigment-Partikel stark gehemmt, sodass das Display ein Bild über Wochen und Monate behält. Zur Veränderung eines Pixels wird ein E-Feld durch die Ansteuerelektroden über die entsprechenden Kapseln gelegt, durch Elektrophorese wandern die Pigmente entlang des Feldes in die gewünschte Richtung. Diesen Vorgang bezeichnet man als Refresh oder Update.

Das E-Feld muss dabei im Zeitverlauf besonderen Anforderungen genügen, die den aktuellen Zustand des Pixels, den Zustand benachbarter Pixel, die Temperatur und weitere Parameter berücksichtigen, sodass aufwendige Wellenformen zur Ansteuerung der Pixel berechnet werden müssen, wofür ein EPD-Controller bzw. EPD-Timing-Controller (im Folgenden auch Display-Controller) benötigt wird[13]. Moderne Controller ermöglichen dabei, auch nur einen bestimmten Anteil der Pigmente durch die Kapsel zu bewegen, wodurch sogar Grausstufen möglich sind. Das komplette weiß- bzw. schwarz-Überschreiben ist hingegen unkritisch, hierzu muss lediglich ein genügend starkes Feld ausreichend lange angelegt werden.

Der Stromfluss bzw. die Leistungsaufnahme des Panels beim Refresh ist dadurch begründet, dass die Partikel gegen den Reibungswiderstand des Polymers durch die Kapseln bewegt werden müssen, d.h. mit steigender Panelgröße steigt die Leistungsaufnahme.

2.3 Energy-Harvesting

2.3.1 Grundlagen Energy-Harvesting

Als *Energy-Harvesting* wird zumeist die Gewinnung von elektrischer Energie aus anderen Energieformen der *unmittelbaren* Umgebung eines (Mikro-) Systems bezeichnet. Es wird dabei nicht nach der Ursache (also natürlich oder künstlich) dieser Energieformen unterschieden, wesentlich ist, dass das Sammeln der Energie tatsächlich am System selbst geschieht[1]. Die obere Hälfte von Abb. 2.3.1 zeigt die dafür infrage kommenden physikalischen Energieformen.

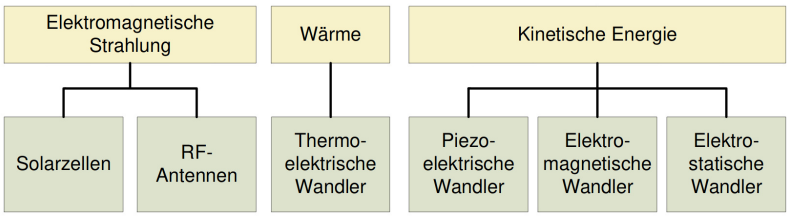


Abbildung 2.3.1: Verfügbare Umgebungsenergiequellen und zugehörige Wandler. [8, S.5]

2.3.2 Energiewandler für Energy-Harvesting

Die untere Hälfte von Abb. 2.3.1 zeigt eine Auswahl möglicher Wandler für die verschiedenen Energieformen. Es sollen nur diese mit möglicher Relevanz für diese Arbeit betrachtet werden.

Wandler für kinetische Energie

Diese sind für das Türschild ohne Relevanz, da dieses fix an einer Wand angebracht wird, und dann über die Lebenszeit nicht mehr bewegt wird. Vernachlässigt man Erdbeben und Zugluft, bleibt der Benutzer als einzige mögliche Ursache für das Freiwerden von kinetischer Energie, im einfachsten Fall z.B. über eine Kurbel am Gehäuse. Dies ist jedoch wenig komfortabel, und tangiert daher Anforderung 2c, außerdem ist zweifelhaft ob diese Art der Energiegewinnung als Energy-Harvesting bezeichnet werden kann.

Wärme

Wärme oder genauer gesagt Temperaturgradienten sind als Energiequelle grundsätzlich denkbar. Ist z.B. eine kalte Mauer vorhanden, könnte das Türschild auf

[1] D.h. im Bezug auf diese Arbeit, dass eine Solarzelle z.B. direkt am Türschild angebracht sein muss, statt über ein Kabel mit einer auf dem Dach befindlichen Solarzelle versorgt zu werden.

der Rückseite mit einem Peltier-Element ausgestattet sein, das die Temperaturdifferenz zwischen Mauer und warmer Umgebungsluft in elektrische Energie wandelt. Dies erfordert jedoch das Vorhandensein einer kalten Mauer, und damit spezielle Bedingungen, wie sie durch Anforderung 1d gerade vermieden werden sollen.

Elektromagnetische Strahlung

Elektromagnetische Strahlung bleibt damit als einzig relevante Energieform. Das Elektromagnetische Spektrum wird zwar zunehmend überfüllter, jedoch sind die Signale bzw. Wellen die z.B. durch W-Lan- und Bluetoothgeräte ausgesandt werden kaum für die Zwecke dieser Arbeit nutzbar. Grund dafür ist die viel zu hohe Freiraumdämpfung von Elektromagnetischen Wellen und die Tatsache dass funkende Systeme ihre Sendeleistung i.d.R. soweit wie möglich reduzieren.

Nahfeldwellen bzw. NFC

Das Türschild soll eine NFC-Konfigurationsschnittstelle aufweisen. Diese besteht im Wesentlichen aus einer Sende- und Empfangsspule, die mit geringem Abstand aufeinander gelegt werden. Der magnetische Wechselfluss der Sendespule induziert in der Empfangsspule eine Spannung, gleich einem Transformator mit Luftkern, sodass im Gegensatz zu Fernfeldwellen tatsächlich nennenswert Leistung übertragen werden kann.

Dies ist das übliche Verfahren zum Versorgen von RFID- und NFC-Karten, z.B. beim neuen Personalausweis, Zugangskarten und Chips, sowie beim Kontaktlosen bezahlen. Hierfür ist an einigen NFC-Transceiver-Bausteinen wie z.B. denen der ST25-Reihe von ST-Microelectronics, ein Harvesting-Pin herausgeführt, der zum Abgreifen der durch das Nahfeld induzierten Spannung bzw. Leistung dient. Fraglich ist aber, ob ein Smartphone tatsächlich genügend Leistung über NFC bereitstellen kann, um das Türschild unter den gegebenen Anforderungen zu versorgen. Dies soll in Absch. 3.2.2 untersucht werden.

Licht

Abgesehen von Nahfeldwellen bleibt Licht damit als einzig denkbare Energiequelle zur Versorgung. Als Energiewandler kommen hier Solarzellen zum Einsatz, denen der folgende Abschnitt gewidmet wird.

2.4 Solarzellen

Hier sollen einige Grundlagen zum Verständnis von Solarzellen erabeitet werden, um zu untersuchen, welche Art von Zelltyp für die vorliegende Anwendung am geeignetsten ist.

2.4.1 Grundlagen

Abb. 2.4.1 zeigt die Bandstruktur einer Solarzelle. Diese entspricht im Wesentlichen der einer pn-Diode. Treffen Lichtquanten passender Energie[2] auf das p-Gebiet, können diese durch Elektronen absorbiert werden, wodurch die Elektronen angeregt und ins Leitungsband gehoben werden. Dort *driften* sie entlang des E-Feldes über der Raumladungszone zum n-Gebiet und können dort über einen externen Verbraucher zurück zum p-Gebiet fließen. Die über dem Verbraucher sichtbare Spannung

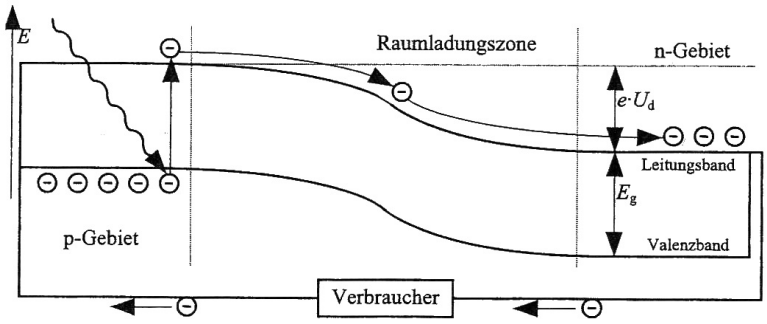


Abbildung 2.4.1: Funktionsprinzip einer Solarzelle im Bändermodell. [9]

entspricht dem Abstand der Unterkante des Leitungsbandes im n-Gebiet zur Oberkante des Valenzbandes im p-Gebiet. Ist der Stromkreis unterbrochen, lädt sich das n-Gebiet gegenüber dem p-Gebiet mehr und mehr auf, bis das entstehende E-Feld das Feld über der Raumladungszone aufhebt. Dann ist die Bandkrümmung aufgehoben, Valenz- und Leitungsband verlaufen gerade, die von außen sichtbare Spannung entspricht der Bandlücke des Halbleiters.

2.4.2 Maximum-Power-Point (MPP)

Wird eine Solarzelle im Leerlauf betrieben, ist ihre Ausgangsspannung $U_\mathrm{Z} = U_\mathrm{LL}$ und maximal. Der durch die Zelle gelieferte Strom I_Z ist null, und damit auch die von der Zelle abgegebene Leistung P_Z. Wird die Zelle dann mit einem Widerstand R_L belastet, sinkt U_Z monoton mit R_L, während P_Z steigt.

[2] Die Energie des Photons $E_\nu = hf$ muss in etwa der Bandlücke des Halbleiters entsprechen, d.h. das anregende Licht muss eine zur Bandlücke passende Wellenlänge $\lambda_\nu \geq 1/f_\nu$ haben.

Im Kurzschlussbetrieb hingegen ist $U_Z = 0$ und I_Z ist maximal, trotzdem ist $P_Z = 0$. Wird R_L nun vergrößert, sinkt der Strom monoton mit steigendem R_L, während P_Z wieder steigt.

Für die Leistung der Solarzelle als Produkt $P_Z = U_Z \cdot I_Z$ gilt damit, dass sie sowohl bei $R_L = 0$, als auch bei $R_L = \infty$ null ist, und dazwischen einen Maximalwert bei U_{MPP} einnimmt. Dieser Punkt auf der $I(U)$-Kurve der Solarzelle wird als Maximum-Power-Point bezeichnet und ist der Betriebspunkt der Solarzelle, an dem diese, die unter den jeweiligen Beleuchtungsbedingungen maximale Leistung abgibt. Für Siliziumsolarzellen liegt dieser Punkt etwa bei $U_{MPP} \approx 0,8 \cdot U_{LL}$.

Weil sich U_{LL} aber mit der Beleuchtungsstärke ändert, implementieren Energy-Harvester und Photovoltaikmodule i.d.R. einen Maxiumum-Power-Point-Tracking-Algorithmus (MPPT), der die Leerlaufspannung der Solarzelle periodisch erfasst, und die Belastung der Solarzelle anschließend so anpasst, dass sich eine Betriebsspannung von $0,8 \cdot U_{LL} = U_{MPP}$ einstellt und die Zelle somit immer die maximale Leistung abgibt.

2.4.3 Typen

Abb. 2.4.2 zeigt eine aktuelle Übersicht (Stand 2019) zur Wandlungseffizienz $\eta = P_{opt.}/P_{el.}$ verschiedener Typen von Solarzellen, die Sich hinsichtlich Aufbau (z.B. einzelne oder mehrere pn-Übergänge), Halbleitermaterial (z.B. Silizium, Galliumarsenit oder organische Zellen) und Halbleiterstruktur (z.B. amorphes oder kristallines Silizium) unterscheiden lassen.

Es wird ersichtlich, dass vor allem Zellen mit mehreren pn-Übergängen und solche aus Galliumarsenit sehr hohe Wirkungsgrade bis über 40% bieten, allgemein ist seit 2010 ein stärkerer Trend nahezu aller Technologien nach oben zu erkennen. Weit ernüchternder ist die kommerzielle Verfügbarkeit der verschiedenen Zelltechnologien, die sich, gegeben die Anforderungen dieser Arbeit (kleine Zellen oder Module mit einer Kantenlänge bis etwa 20 cm), auf amorphe und monokristalline Siliziumzellen in Dünnschichttechnologie reduziert. Selbst hier ist die Auswahl nicht besonders groß, es scheint als gäbe es (noch) keinen ausreichenden Bedarf an hocheffizienten, kleinen Modulen.

2.4.4 Vergleich der Typen im Innenbereich

Aufgrund der Abhängigkeit der Effizienz einer Solarzelle von der Wellenlänge des auftreffenden Lichts wird zum Vergleich verschiedener Zellen ein standardisiertes Sonnenspektrum (AM-1.5) verwendet. Dieses ist für Innenräume wenig geeignet, da

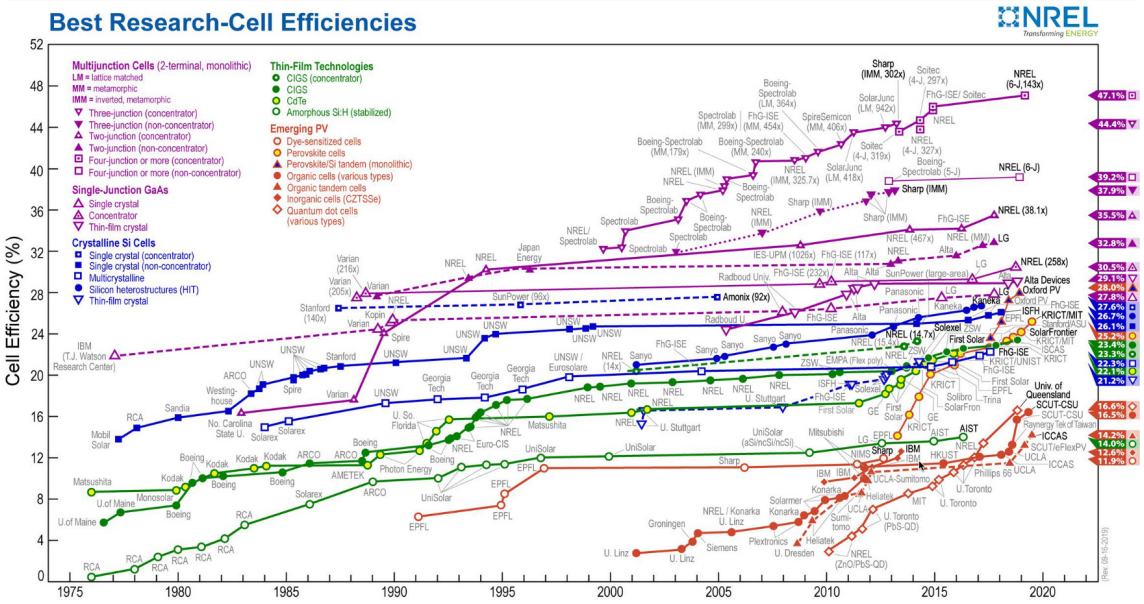


Abbildung 2.4.2: Wandlungseffizienz $\eta = P_{\text{opt}}/P_{\text{el}}$ verschiedener Typen von Forschungs-Solarzellen nach Erscheinungsjahr.[10]

hier auch die spektralen Anteile künstlicher Lichtquellen wie Leuchtstoffröhren und LED-Lampen berücksichtigt werden müssen.

Abb. 2.4.3 zeigt die externe spektrale Quantenausbeute verschiedener Solarzelltechnologien. Klassische monokristalline Siliziumzellen (c-Si) absorbieren sehr gut, bis weit über den sichtbaren Bereich des Spektrums (380 nm − 750 nm) hinaus. Daher domnieren diese im Bereich der Outdoor-Siliziumzellen, weil diese mehr der auftreffenden Photonen absorbieren als z.B. amorphe Siliziumzellen (a-Si). Da künstliche Lichtquellen aber idealerweise nur im sichtbaren Bereich strahlen, ist zu erwarten, dass Zellen die nur in diesem Bereich absorbieren unter reinen Indoorbedingungen effizienter sind.

Abb. 2.4.4 bestätigt dies. Es zeigt sich, dass amorphe Siliziumzellen unter Kunstlicht erheblich effizienter sind als monokristalline Siliziumzellen.

2.5 Energiespeicher

Als Energiespeicher kommen grundsätzlich Super-Caps und Akkumulatoren infrage. Der Energiespeicher dient als Puffer, da das Türschild voraussichtlich am häufigsten Vormittags benutzt wird, sich aber über den ganzen Tag verteilt auflädt. Demnach sollte der Energiespeicher so groß sein, dass z.B. mindestens 10 Updates bereitstehen, d.h. er soll eine Energie von mindestens 45 Ws vorhalten (s.Absch. 3.1.2).

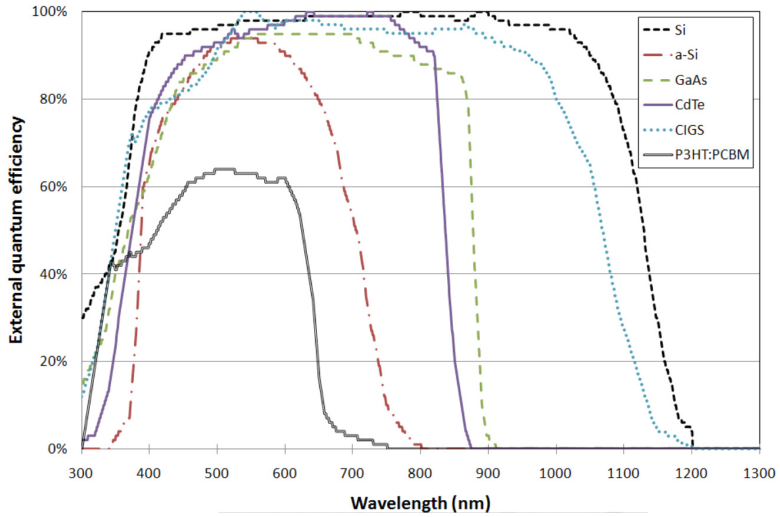


Abbildung 2.4.3: Spektraler externer Quantenwirkungsgrad verschiedener Zelltypen.[11]

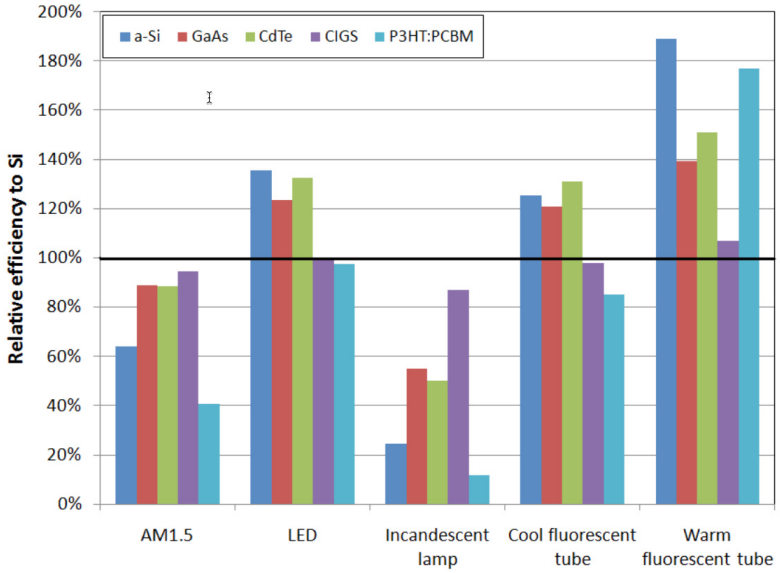


Abbildung 2.4.4: Relative Effizienz verschiedener Zelltypen im Vergleich zu c-Silizium bei verschiedenen Lichtspektren.[11]

2.5.1 Super Caps

Super-Caps und ihr Superlativ Ultra-Caps sind Kondensatoren mit sehr hohen Kapazitäten in der Größenordnung ab $0,1\,\mathrm{F}$ aufwärts. Technologisch sind Sie eine Weiterentwicklung der Doppelschichtkondensatoren, es gibt aber auch Super-Caps, die Energie elektrochemisch binden, oder Hypride der beiden Technologien. Sie weißen zwar geringere Energiedichten als z.B. Li-Ion-Akkus auf, jedoch können sie höhere Lade- und Entladeleistungen umsetzten. Die Energie eines Kondensators ist $E = 1/2\,C\,U^2$, demnach beträgt die notwendige Kapazität bei einer optimistischen Spannung von z.B. $5\,\mathrm{V}$

$$C = 2\,E\ /\ U^2 = 2 \cdot 45\,\mathrm{Ws} \cdot (5\,\mathrm{V})^2 = 3,6\,\mathrm{F}$$

was durchaus realistisch erscheint. Problematisch ist aber die hohe Selbstenladung von Super-Caps[3]. Beim `BCAP0003 P300 X11` von Maxwell Technologies, einem $3\,\mathrm{V}$ Super-Cap mit $3\,\mathrm{F}$ liegt diese bei $7\,\mu\mathrm{A}$[14], bei der $50\,\mathrm{F}$ Variante sogar bei $100\,\mu\mathrm{A}$[15].

2.5.2 Li-Ion- und Li-Po-Akkumulatoren

Lithium-Ionen-Akkus und (Li-Ion-Akkus) und Lithium-Ionen-Polymer-Akkus (Li-Po-Akkus) binden Energie elektrochemisch unter Beteiligung von Lithium-Li^+-Ionen. Lithium bietet mit $E_0 = -3,04\,\mathrm{V}$ das höchste Standardpotenzial der Elektrochemischen Spannungsreihe[16, S. 356, Tab. 3.13], was die hohe Energiedichte und Nennspannung der Akkus von $3,7\,\mathrm{V}$ erklärt[17, S. 9ff.]. Li-Po-Akkus sind eine Subgruppe der Li-Ion-Akkus, gekennzeichnet durch die Verwendung eines gelartigen Elektrolytes auf Polymerbasis, was u.a. flache Bauweisen wie die der *Pouch*-Zelle ermöglicht.

Beide Akkus werden nach wie vor zunehmend beliebter und bieten immer höhere Energie- und Leistungsdichten, getrieben durch die rasant fortschreitende Verbreitung von mobilen-IT-Endgeräten (Smartphones, Tablets, Laptops), wo hauptsächlich Li-Po-Pouch-Zellen verwendet werden, und Elektromobilität (E-Bikes, Elektroautos). Beide Anwendungsgebiete erfordern Akkus mit hoher Leistungsdichte, die ohne Memory-Effekt und bei geringer Degradierung hohe Ladeströme tolerieren.

Wichtigste Kenngröße für diese Arbeit ist die Selbstentladung. Diese liegt je nach Quelle zwischen $2\% - 10\%$[18] der ursprünglichen Kapazität, pro Monat. Bei einer

[3] Die nur einer von 5 betrachteten Anbietern im Datenblatt angegeben hat.

Selbstenladerate von 2% verliert ein geladener Akku mit 2000 mAh damit pro Monat 20 mAh an Kapazität, er entläd sich in dieser Zeit mit $20\,\mathrm{mAh}/(30,5 \cdot 24\mathrm{h}) = 27\,\mu\mathrm{A}$. Ein Akku mit einer Kapazität von nur 100 mAh hingegen entläd sich entsprechend mit nur $1,37\,\mu\mathrm{A}$.

Die Energie im letztgenannten Akku liegt bei Nennspannung $3,6\,\mathrm{V}$ in der Größenordnung von $3,6\,\mathrm{V} \cdot 100\,\mathrm{mAh} \cdot 3600\,\mathrm{s} \cdot \mathrm{h}^{-1} = 259\,\mathrm{Ws}$ und damit wesentlich höher als die in Super-Caps mit vergleichbarer Selbstentladung speicherbare Energie.

Kapitel 3

Systementwurf und Komponentenauswahl

3.1 Auswahl Display

Gemäß Anforderung 2b muss das Display des Türschildes ausreichend groß sein, als Richtwert wurde die Größe der Türschilder in der Hochschule Offenburg verwendet, also etwa 15 cm × 15 cm. Generell sind EPDs dieser Größe zum aktuellen Zeitpunkt schwer in geringen Stückzahlen beschaffbar.

Ein Anbieter solcher Displays ist Waveshare, der sich eher an Privatkunden richtet. Es werden zahlreiche EPDs unterschiedlicher Größe angeboten, zusammen mit einem jeweils passenden Controller-Board. Weiterhin ist viel Beispielcode zu den Displays vorhanden, außerdem sind die Displays auf Wiki-Seiten gut dokumentiert.

3.1.1 Waveshare 7.8” und 9.7” E-Paper-Displays

Infrage kommen die Displays `7.8inch e-Paper HAT` und `9.7inch e-Paper HAT`[1]. Die Daten der beiden Displays sind in Tab. 3.1 dargestellt, entnommen aus [19] und [20].

Das 7,8 ”-Display hat bei geringerer Größe eine höhere Auflösung, da die Pixel nur halb so groß sind. Damit ist auch die Leistungsaufnahme während eines Refreshs größer, laut Hersteller um Faktor 2. Diese Werte sind jedoch kritisch zu hinterfragen, da die Leistungsaufnahme während eines Updates stark schwankt, die Angabe des Energieverbrauchs wäre daher wesentlich sinnvoller.

[1] Die Abkürzung `HAT` steht für *H(ardware) A(ttached) (On) T(op)*. und bedeutet, dass das Controller-Board Pinkompatibel zu der 40-Pin-Stiftleiste auf einem Rasberry Pi ist.

Tabelle 3.1: Herstellerangaben zu den 7,8 " und 9,7 " EPDs von Waveshare.

Eigenschaft	**7.8inch e-Paper HAT**	**9,7inch e-Paper HAT**
Abmaße L×B×H:	173,8 mm×127,6 mm×0,78 mm	218,8 mm×156,425 mm×1,15 mm
Auflösung:	1872 px×1404 px	1200 px×825 px
Pixelgröße:	84,5 µm×84,5 µm	169 µm×169 µm
Anzahl Graustufen:	2-16	2-16
Refreshdauer:	450 ms	$<$ 1 ms
Leistungsaufnahme Refresh:	1200 mW	600 mW
Leistungsaufnahme Standby:	100 mW	300 mW

Für beide Displays wird ein nahezu identisches Controllerboard geliefert, jeweils bestückt mit einem `IT8951`-Timing-Controller. Dieser verfügt zwar über einen Sleepmode, jedoch benötigt das Controllerboard laut Messung auch hier etwa 50 mW Leistung.

Damit ist es zwingend notwendig das Controllerboard und das Display über einen Loadswitch von der Spannungsversorgung zu trennen und nur für das Update einzuschalten. Weil hierbei der Inhalt des flüchtigen Imagebuffers des Controllers verloren geht, ist es nötig das Display dann zunächst weiß zu überschreiben (clear-white). Es genügt nicht den Imagebuffer komplett mit dem neuen Bild zu überschreiben, da der Controller für die Berechnung der Wellenform wissen muss, wie das aktuelle Bild auf dem EPD aussieht. Lediglich das Überschreiben mit weiß oder schwarz ist auch ohne validen Inhalt des Imagebuffers möglich. Danach ist der Buffer in einem definierten validen Zustand und das Display kann mit dem Bild überschrieben werden.

Bleibt der clear-white vor einem Update aus, nachdem die Versorgungsspannung getrennt wurde, wird das Bild wie in Abb. 3.1.1 gezeigt, fehlerhaft dargestellt.

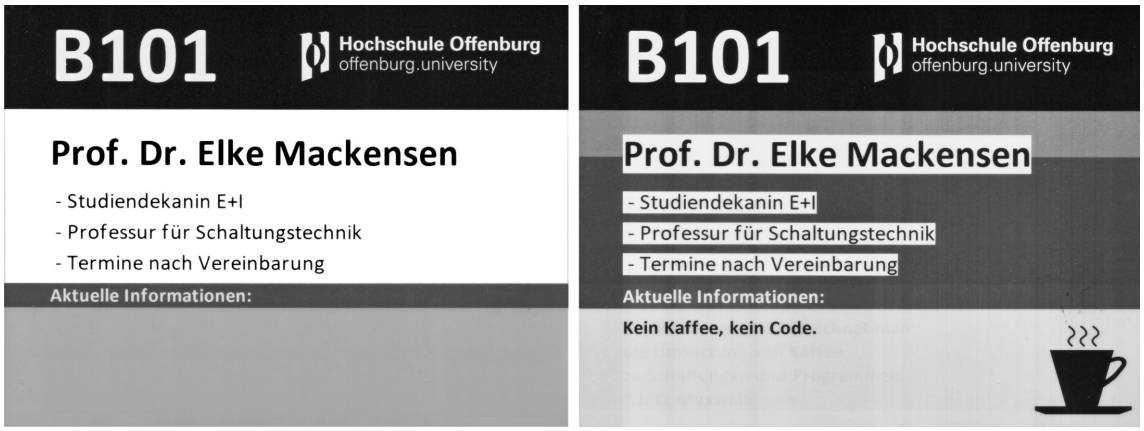


Abbildung 3.1.1: Display vor und nach Update ohne clear-white nach Trennung der Versorgungsspannung.

Aufgrund der für Büro-Türschilder passenderen Größe und der besseren Auflösung wurde das 7,8 ”-Display gewählt, auch wenn die Leistungsaufnahme laut Hersteller doppelt so hoch ist. Zur Abschätzung des Energieverbrauchs wurde das 7,8”-Display daher messtechnisch analysiert.

3.1.2 Energieverbrauch des 7,8”-EPD

Zur Messung des Energieverbrauchs wurde das Controller-Board aus einer Source-Measure-Unit (SMU) versorgt und die Eingangsleistung periodisch alle 150 µs gesampelt. Der Controller befand dabei zunächst im Sleepmode, wurde dann initialisiert und führte den clear-white aus. Dann wurde ein Bild auf das Display geschrieben, bevor der Controller wieder in den Schlafmodus wechselte.

Leistungsaufnahme $P[n]$ und Energieverbrauch $E[n]$ sind im diskreten Zeitbereich über die kumulative Summe

$$\begin{aligned} E[n] &= \sum_{i=0}^{n} \left(P[i] \cdot 150\,\mu\text{s}\right) \\ &= 150\,\mu\text{s} \sum_{i=0}^{n} P[i] \end{aligned}$$

miteinander verknüpft. Wegen der hohen Zeitauflösung gegenüber der Updatedauer werden $P[n]$ und $E[n]$ im Folgenden als quasikontinuierliche $P(t)$ und $E(t)$ angenommen und notiert.

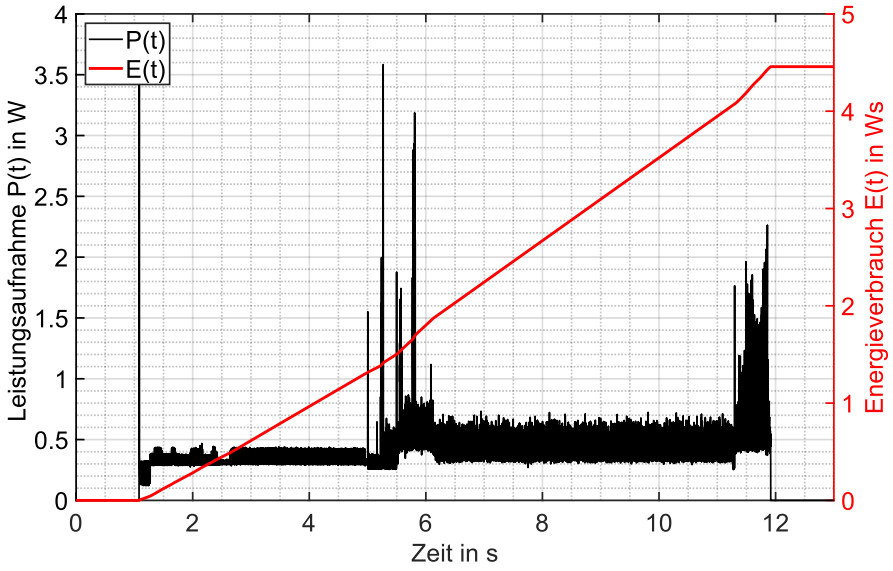


Abbildung 3.1.2: Leistungsaufnahme und Energieverbrauch während eines Displayupdates mit vorherigem Clear.

In Abb. 3.1.2 sind die Messergebnisse dargestellt. Gezeigt ist der Auschnitt, in dem der Controller wach, d.h. nicht im Sleepmode ist. Damit zeigt der Ausschnitt den minimal möglichen Bereich, indem das Controllerboard mit Strom versorgt werden muss.[2]

Es zeigt sich, dass ein Update einen Energieverbrauch von $4,5\,\mathrm{Ws}$ verursacht und dass ein Update mit vorherigem clear-white knapp 11 s dauert. Die Spitzenleistung beträgt über 2.5 W, sodass die Energieversorgung des EPD-Controllers ausreichend niederimpedant sein muss. Die Peaks von P(t) ab $t = 4\,\mathrm{s}$ sind durch den tatsächlichen Refresh des Displays auf weiß verursacht, die Peaks ab $t = 10$ durch den Refresh auf das eigentlich darzustellende Bild. Die Leistungsaufnahme beträgt nahezu durchgehend über 300 mW und deckt sich mit der Angabe `Power Dissipation: 300 mW` im Datenblatt des `IT8951`-Controllers[21]. Weitere Angaben zu Stromverbrauch bzw. Leistungsaufnahme finden sich dort nicht.[3]

3.2 Auswahl NFC-Transceiver

3.2.1 `ST25DV`-Transceiver

Als NFC-Transceiver Baustein wurde der `ST25DV`-Transceiver gewählt. Dieser wurde mit guten Erfahrungen auch von der SKEID-Gruppe eingesetzt. Er verfügt über eine RF-Schnittstelle bei $13,56\,\mathrm{MHz}$ und erfüllt die Spezifikation eines NFC-Forum-Type-5-Tag bei maximalen Datenraten bis 53 kb/s.

Zusätzlich ist eine I^2C-Schnittstelle zur Konfiguration und zum Auslesen der NFC-Nachrichten mit einer Datenrate bis 1 Mb/s vorhanden.

Bei einem Preis von unter 1€ ist der Transceiver äußerst kostengünstig und aufgrund einiger weiterer Eigenschaften sehr gut für den Einsatz im Türschild geeignet:

1. **Mailbox**
   Der `ST25DV` ist mit einer Mailbox ausgestattet, in die Daten über die RF-Schnittstelle vom NFC-Reader abgelegt werden können. Diese können dann von einem Mikrocontroller über die I^2C-Schnittstelle ausgelesen werden. Ebenso kann der Controller über die I^2C-Schnittstelle Daten in der Mailbox ablegen, die dann vom Reader über die NFC-Schnittelle ausgelesen werden können.

[2] Genaugenommen muss der Controller nach Anlegen der Spannung über den SPI-Bus neu initialisiert werden, was beim Aufwachen wegfällt. Diese Zeit ist gegenüber der Updatedauer aber vernachlässigbar.

[3] Allgemein ist das Datenblatt unbefriedigend. Es konnte z.B. nicht herausgefunden werden konnte, wie der SD-RAM des Controllers zu beschreiben oder auszulesen ist.

Liegt eine ungelesene Nachricht in der Mailbox, ist diese für eine einstellbare Zeit schreibgeschützt, bis die Nachrichten ausgelesen werden.

Die Mailbox stell damit eine besonders einfache Art der Umsetzung von Nachrichten zwischen den beiden Schnittstellen dar.

Die RF-Seite der Mailbox wird direkt aus einem NFC-Feld versorgt, somit werden NFC-Nachrichten auch in der Mailbox abgelegt, wenn der `ST25DV` nicht mit Spannung am V_{CC}-Pin versorgt wird.

2. **GPO-Interrupt-Pin (GPO-Pin)**
   Der Transceiver verfügt über einen Interrupt Ausgang, auf dem programmierbare Interrupts ausgegeben werden, z.B. wenn der Tag ein RF-Feld detektiert, wenn ein vorhandenes Feld verschwindet oder wenn sich der Status der Mailbox(nachrichten) ändert, also z.B. eine Nachricht in die Mailbox geschrieben wurde.

3. **Energy-Harvesting-Pin (EH-Pin)**
   Es ist ein Energy-Harvesting-Pin vorhanden, an dem die durch den Transceiver aus dem NFC-Feld geerntete Spannung bzw. Leistung abgegriffen werden kann, z.B. zur Versogrung externer Komponenten oder des gesamten Systems.

Es wurde daher der `ST25DV1` als NFC-Transceiver ausgewählt, mit dem weiteren Vorteil, dass sich ein Großteil des SKEID-Codes nach Portierung der Low-Level-I^2C-Treiber wiederverwenden ließ. Da der IC über einen EH-Pin verfügt, stellt sich die Frage, ob sich aus diesem in Verbindung mit einem Smartphone eine relevante Energiemenge ernten lässt. Da keine Literatur hierzu gefunden werden konnte, wurde der im nächsten Abschnitt vorgestellte Versuch durchgeführt.

3.2.2 Maximale Belastbarkeit eines Smartphone-NFC-Feldes

Um die Leistung P_{EH}, die sich aus dem EH-Pin ernten lässt, während das NFC-Feld durch ein Smartphone erzeugt wird, wenigstens abschätzen zu können, wurde der EH-Pin eines `ST25DV_Discovery_ANT_C1`-Development-Boards mit verschiedenen Lastwiderständen R_{L} belastet, während die freigelegte Antenne eines Huawei Mate-7 Smartphones in möglichst geringem Abstand und möglichst deckungsgleich auf der Tag-Antenne platziert wurde. Das Smartphone pollt regelmäßig nach einem Reader wenn NFC aktiviert ist, nach Detektion erhält es das NFC-Feld unmoduliert aufrecht, um den Reader mit Energie zu versorgen. Gemessen wurde dann die Spannung am EH-Pin U_{EH}.

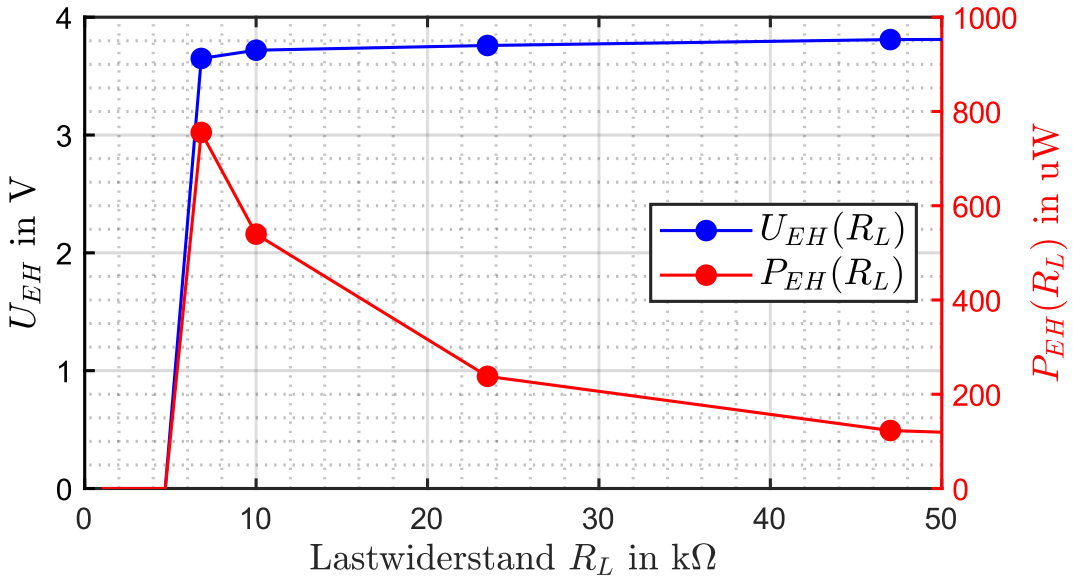


Abbildung 3.2.1: Spannung und Leistung am EH-Pin in Abhängigkeit vom Lastwiderstand R_L, wenn als Reader ein Smartphone eingesetzt wird.

Das Messergebnis ist in Abb. 3.2.1 dargestellt. Erwartungsgemäß steigt die Leistung mit sinkendem R_L, während die Spannung nahezu unverändert $3,75\,\mathrm{V}$ beträgt. Etwa bei $R_\mathrm{L} = 6,8\,\mathrm{k\Omega}$ erreicht P_EH den Maximalwert von $750\,\mu\mathrm{W}$. Bei $R_\mathrm{L} = 4,7\,\mathrm{k\Omega}$ ist die Spannung zusammengebrochen, weil das Smartphone das NFC-Feld abgeschaltet hat. Dieses Verhalten war reproduzierbar, daher ist davon auszugehen, dass das Huawei Mate-7 keine höheren Belastungen des NFC-Feldes toleriert.

Unter optimalen Bedingungen würde es somit $4,5\,\mathrm{Ws}/750\,\mu\mathrm{W} \approx 1:40\,\mathrm{h}$ dauern, bis genug Energie für *ein* Update geerntet wäre, also untolerierbar lange. Angenommen die NFC-Kommunikation würde z.B. $5\,\mathrm{s}$ dauern, könnten in dieser Zeit $3,75\,\mathrm{mWs}$ geerntet werden, also weniger als ein tausendstel der Updateenergie. Daher wurde der Ansatz, den EH-Pin zur Stromversorgung zu nutzen verworfen.

3.3 Auswahl Solarzelle

In Absch. 2.4.4 wurde festgestellt, dass amorphe Siliziumzellen unter Kunstlicht effizienter sind, als monokristalline Zellen. Um dies zu bestätigen und um eine Größenordnung für die durch eine Solarzelle gelieferte Leistung unter schwacher Beleuchtung, wie in den Gängen der Hochschule Offenburg, zu erhalten, wurde die im nächsten Abschnitt vorgestelte Messreihe durchgeführt.

3.3.1 Messreihe mit verschiedenen Zelltypen

Zur abschließenden Auswahl der Zelltechnologie wurde eine Testreihe im Innenbereich gefahren, bei der die $P(I)$-Kennlinien dreier verschiedener amorpher und monokristalliner Zellen drinnen bei sehr schwacher Beleuchtung unter überwiegend Kunstlicht (warm-weißes-Leuchtsoffröhrenlicht), Mischlicht und indirektem Sonnenlicht, d.h. Streulicht aufgenommen wurde. Hierzu wurden die Zellen nebeneinander platziert und nacheinander in kurzer Folge mit einem Poti belastet und die Spannung über und der Strom durch das Poti mittels Multimeter gemessen. Daraus wurde die im Poti umgesetzte Leistung berechnet, die durch Beleuchtungsstärke und Fläche der Zelle geteilt wurde, um eine *flächen- und beleuchtungsbezogene* Größe zu erhalten. Die Beleuchtungsstärken sind durchaus repräsentativ für schwach bis normal beleuchtete Gänge in der Hochschule Offenburg.

Getestet wurden die `AM-1815`, eine amorphe Dünnschicht-Siliziumzelle für Indooranwendungen aus der Panasonic-Amorton-Reihe, die `SLMD121H10L` von IXYS, eine monomonokritalline Zelle für Indoor und Outdooranwendungen, sowie eine preisgünstige Zelle mit Bezeichnung `0.5W Solar Panel 55x70` von Seeed.

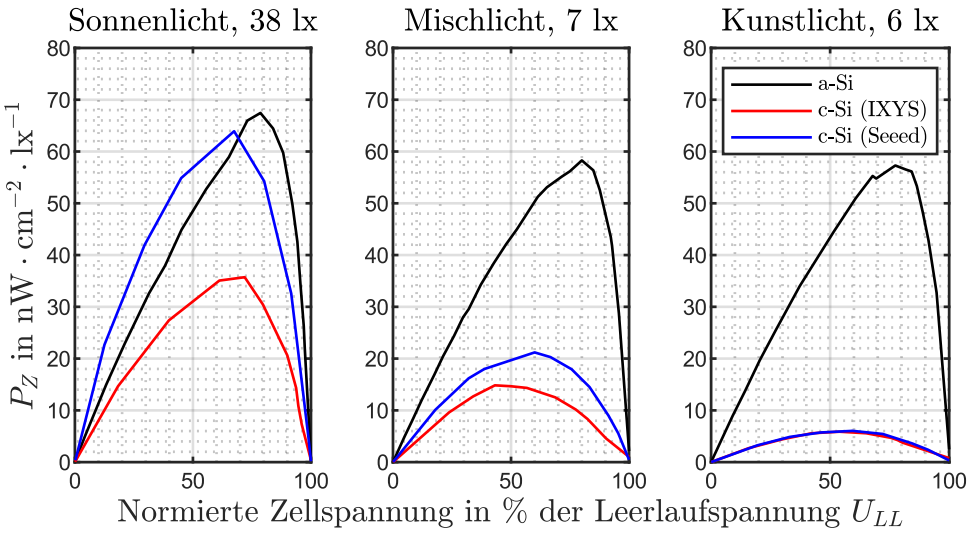


Abbildung 3.3.1: Flächenbezogene Ausgangsleistung pro Lux bei verschiedenen Beleuchtungssituationen über der Zellspannung.

Wie in Abb. 3.3.1 ersichtlich, dominiert in allen drei Fällen klar die amorphe Zelle. Der MPP liegt hier erwartungsgemäß bei etwa 80% der Leerlaufspannung. Die monokristallinen Zellen unterscheiden sich umso stärker, je größer der Anteil des Sonnenlichtes im Spektrum wird. Erstaunlicherweise ist die Effizienz der preisgünstigen Seeed-Zelle sogar höher als die der IXYS-Zelle.

Gegeben die Beleuchtungssituation bietet die amorphe Zelle die mit Abstand größte Energieausbeute. Damit wurde im System eine Indoor Zelle aus der Amorton-

Reihe von Phillips verwendet. Diese sind in verschiedenen Größen und Leerlaufspannungen gut verfügbar. Der ausgewählte Harvester-IC (s. Absch. 3.4.2) benötigt eine Eingangsspannung $U_{\text{in,min}} \leq 2,6\,\text{V}$. Damit wurde mit der `AM-1454` die größte beschaffbare Zelle mit passender Leerlaufspannung der Amorton Serie ausgewählt. Tab. 3.2 zeigt die Daten[22].

Tabelle 3.2: Daten der amorphen Dünnschichtzelle Amorton AM-1454.

Eigenschaft	**Wert**
Leerlaufspannung V_{LL} bei 200 lx	2,5 V
Kurzschlussstrom I_{KS} bei 200 lx	35,2 µAh
Abmaße (L×B×H)	41,6 mm×26,3 mm×1,1 mm
Fläche	10,94 cm^2

Bei Verwendung von 20 Zellen ergibt sich eine Fläche von $218,5\,\text{cm}^2$. Unter ungünstigster Beleuchtung von z.B. 6 lx ständen nach obigen Messungen 60 nW cm^{-2} $\text{lx}^{-1} \cdot 6\,\text{lx} \cdot 210\,\text{cm}^2 = 75,6\,\mu\text{W}$ zur Verfügung, bei einer Spannung am MPP von $0,8 \cdot 2,5\,\text{V}$ entsprechend $37,8\,\mu\text{A}$. Damit ließe sich das Display maximal 1,45 mal pro Tag updaten.

3.3.2 Beleuchtungsstärken

Um die Beleuchtungsstärken besser einschätzen zu können, sind in Abb. 3.3.2 zwei Beleuchtungssituationen dargestellt. Diese wurden Abends nach der Dämmerung in der Hochschule Offenburg aufgenommen und sind daher als Worst-Case-Beleuchtungen zu werten.

Abbildung 3.3.2: Beispiel zweier schlechter Beleuchtungssituationen.

3.3.3 Fazit Auswahl Solarzelle

Amorphe Siliziumzellen weißen laut Literatur und eigenen Messungen bei schlechten Beleuchtungsbedinungen unter 50 lx sowohl unter Kunst- als auch Sonnenlicht die beste Effizienz auf. Als Solarzelle wurde die amorphe Siliziumzelle Amorton `AM-1454` gewählt. Bei Verwendung von 20 Zellen werden eine Ausgangsleistung von $75,6\,\mu A$ und entsprechend maximal 1,45 Updates pro Tag erwartet.

3.4 Auswahl sonstige Komponenten

3.4.1 Auswahl Energiespeicher

Wie in Absch. 2.5 festgestellt ist die Selbstentladerate von Super-Caps geeigneter Kapazität für die vorliegende Anwendung zu hoch. Weiter wurde festgestellt, dass der Selbstentladestrom von Li-Po-Akkus mit der Kapazität der Zelle steigt. Dies ist bei *aufladbaren* Geräten irrelevant, denn die Selbstentladerate ist im Gegensatz zum Selbstentladestrom definitionsgemäß unabhängig von der Kapazität.

Hier aber dient der Akku nur als Puffer, sodass der Selbstenladestrom dauerhaft durch die Solarzelle geliefert werden muss. D.h. dass der Akku grundsätzlich möglichst klein sein sollte, damit der Selbstentlasestrom möglichst klein ist.

Es wurde sich damit für den Li-Po-Akku `ASR00011` von TinyCircuits in Pouchform entschieden. Tab. 3.3 zeigt die relevanten Daten.

Tabelle 3.3: Daten des gewählten Li-Po-Akkus `ASR00011`[1, S. 1].

Eigenschaft	**Wert**
Nennspannung	3,7 V
Kapazität	70 mAh
Ladeschlussspannung	4,2 V
Entladeschlussspannung	3,0 V
Abmaße (L × B × H)	16 mm× 15 mm× 5 mm
Selbstentladestrom (bei 1% Selbstentladungsrate)	1,91 µA

Der gewählte Akku kann damit eine Energiemenge von etwa $3,7\,V{\cdot}70\,mAh{\cdot}3600s{\cdot}h^{-1} = 932\,Ws$ speichern, also genug Energie für 200 Updates. Weiterhin verfügt der Akku über eine integrierte Schutzbeschaltung was zumindest für die Entwicklungs- und Testphase sicherlich sinnvoll ist.

3.4.2 Auswahl Energy-Harvester

Der Energy-Harvester hat zwei Funktionen: Das MPPT für die Solarzelle und das Laden des Energiespeichers. Als weitere Anforderungen sollte der Wirkungsgrad möglichst hoch sein, der Ruhestrom möglichst gering, da dieser wenn die Solarzelle keinen Strom liefert den Akku belastet. Tab. 3.4 zeigt eine Auswahl von Harvester-ICs.

Tabelle 3.4: Auswahl einiger Harvester-ICs[2],[3],[4].

Eigenschaft	**BQ25505**	**SPV1040**	**SPV1050**
Topologie	Boost	Boost	Buck-Boost
Ruhestrom im Betrieb I_q	0,4 µA	60 µA	1,7 µA
Parameter Harvester:			
Max. Eingangsspannung $U_{in,max}$	$\leq U_{Bat}$	$\leq U_{Bat}$	18 V
Min. Eingangsspannung $U_{in,min}$	0,6 V	0,3 V	2,6 V
Min. Eingangsstrom $I_{in,min}$	—	—	5 µA
Min. Eingangsleistung $P_{in,min}$	15 µW	—	13 µA
Parameter Laderegler:			
Max. Akkuspannung U_{bat}	5,1 V	5,2 V	5,3 V
Max. Ladestrom I_{bat}	230 mA	—	70 mA
Max. Ladeleistung P_{bat}	—	3 W	—

Der `SPV1040` ist interessant weil er die kleinste Eingangsspannung benötigt, jedoch ist sein Ruhestrom viel zu hoch. Der `SPV1050` ist als Buck-Boost-Topologie aufgebaut, d.h. er kann die Eingangsspannung sowohl hoch- als auch herabsetzen. Damit muss die von der Solarzelle gelieferte Spannung lediglich zwischen $U_{\text{in,min}} = 2,6\,\text{V}$ und $U_{\text{in,max}} = 18\,\text{V}$ liegen.

Der `BQ25505` hingegen ist als Boost-Topologie aufgebaut, er kann die Eingangsspannung nur hochsetzen. Die maximal von der Solarzelle gelieferte Spannung muss laut Datenblatt[2, S. 8,"`VDELTA`"] 400 mV unter der Entladeschlussspannung 3 V liegen, also unter 2, 6 V.

Da der `BQ25505` von Texas Instruments den kleinsten Ruhestrom benötigt und die minimale Eingansspannung des `SPV1050` mit $U_{\text{in,min=2,6 V}}$ relativ hoch ist, wurde der `BQ25505` gewählt. So ist sichergestellt, dass auch bei schlechter Beleuchtung, wenn die Zellspannung relativ klein ist, der Ladebetrieb sichergestellt werden kann. Die niedrigere Eingansspannung verringert bei konstanten Leckströmen die Verlustleistung, was die Effizienz erhöht. Weiterhin bietet der `BQ25505` die Möglichkeit zusätzlich eine Batterie bzw. Primärquelle anzuklemmen, was für Erweiterungen des Systems nützlich sein könnte.

3.4.3 Auswahl Mikrocontroller

Die SKEID-Gruppe wählte ein `MSP-EXP430FR5994`-Development-Board[23] mit einem `MSP430FR5994`, 16-bit-Mikrocontroller, mit bis zu 16 MHz Taktrate, 256 KiB FRAM-Programmspeicher und 8 KiB SRAM[24]. Dieser bringt jedoch zwei Nachteile mit sich:

- In dieser Arbeit wurde ein Display mit wesentlich höherer Auflösung und mehreren Graustufen gewählt, somit steigt der Speicherbedarf für Piktogramme und Schriften, die beide als Bitmap-Arrays im Programmspeicher liegen, stark an. Mikrocontroller der MSP430-Reihe sind aber nur bis 512 KiB Programmspeicher erhältlich.
- Die CPU mit 16 MHz Taktrate wird den SPI-Bus zum Display nicht mit 24 Mb/s auslasten können.

Weil ein externe Speicher aufgrund des zusätzlichen Energiebedarfs und aufwändigeren Programmierung nicht infrage kam, sollte hier ein MSP432-Mikrocontroller verwendet werden. Dieser enthät einen Cotrex-M4F-Kern, die Peripherie ist jener der MSP430-Mikrocontroller sehr ähnlich, die Blockschaltbilder sind größtenteils identisch, ebenso die Register. Dies bringt den Vorteil, dass ein Großteil des Low-Level-SKEID-Codes wie I^2C-, SPI- und UART-Teiber leicht protierbar sind. Tab. 3.5 zeigt einen Vergleich zweier infrage kommender Mikrocontroller[25],[26] mit dem `MSP430FR5994`.

Tabelle 3.5: Vergleich der MSP430 MCU mit zwei MSP432 MCUs.

Eigenschaft	**MSP430 FR5994**	**MSP432 P4011**	**MSP432 E401Y**
Programmspeicher	256 KiB FRAM	2048 KiB Flash	1024 KiB Flash
RAM	8 KiB SRAM	256 KiB SRAM	256 KiB SRAM
Max. Taktrate	16 MHz	48 MHz	120 MHz
Aktiv-Verbrauch bei max. Taktrate	1,89 mA/	4,8 mA	40 mA
Kleinster Tiefschlaf-Verbrauch	45 nA	22 nA	1,20 µA

Aufgrund des extrem geringen Verbrauchs im Tiefschlafmodus von nur 22 nA wurde die `MSP432P4011`-MCU gewählt. Die Taktrate ist mit 48 MHz zwar nur doppelt so groß wie der SPI-Takt, sodass dieser womöglich nicht voll ausgelastet werden kann, jedoch wird dies aufgrund des mehr als 50-fach niedrigeren Tiefschlafverbrauchs, sowie des doppelt so großen Programmspeichers in Kauf genommen.

3.5 Systemkonzept

Auf Basis der gewählten Komponenten wurde das in Abb. 3.5.1 dargestellte Harware-Systemkonzept erstellt. Dieses wird im Folgenden erläutert.

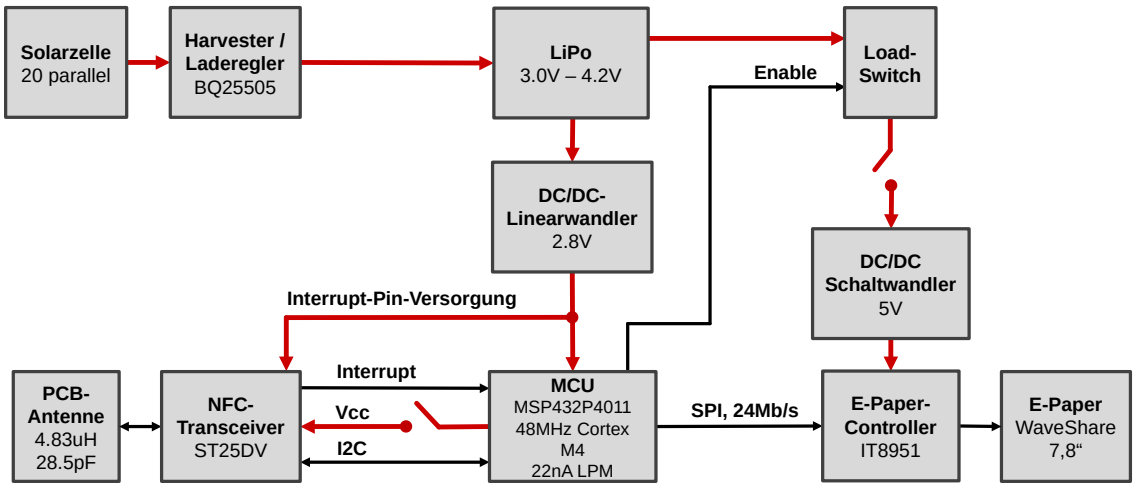


Abbildung 3.5.1: Systemkonzept des Hardware-Entwurfs.

3.5.1 Energiemanagement

Der Energy-Harvester-IC erntet die von der Solarzelle gelieferte Leistung und lädt damit einen LiPo-Akku, dessen Spannung zwischen 3 V und 4, 2 V liegt. Der `MSP432` Mikrocontroller soll aus einem LDO mit 2, 8 V versorgt werden. Im Ruhezustand ist die MCU im Tiefschlaf und benötigt sehr wenig Leistung. Die Stromversorgung des `ST25DV`-NFC-Transceivers ist abgeklemmt, lediglich der Pin für die Versorgung des Interrupt-Pin-Treibers steht unter Spannung und benötigt ebenfalls sehr wenig Leistung.

Der Boost-Wandler für die Versorgung des EPD-Controller-Boards und des EPD-Displays wird über einen Load-Switch direkt aus dem Akku versorgt und wird durch den Switch im Ruhezustand ebenfalls spannungsfrei geschaltet. Damit wird der Akku im Ruhezustand lediglich durch den LDO, die MCU im Tiefschlaf und den Interrupt-Pin des NFC-Transceivers belastet.

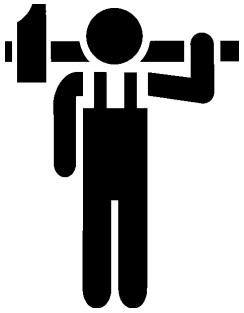
3.5.2 Benutzerinteraktion

Möchte der Benutzer ein Displayupdate vornehmen, hält er das Smartphone gegen die Antenne. Der NFC-Transceiver wird über das NFC-Feld versorgt, empfängt eine erste Nachricht und löst einen Interrupt an der MCU aus.

Diese wacht auf, schaltet die Versorgung des NFC-Transceivers ein, und initialisiert diesen anschließend, damit die erste Nachricht über den I^2C-Bus aus der Mailbox des `ST25DV`-Transceivers ausgelesen werden kann. Damit kann eine neue Nachricht empfangen werden, die wieder ausgelesen wird. Ist die letzte Nachricht empfangen, wird die Versorgung des NFC-Transceivers abgeschaltet.

Die Nachrichten werden in einen Buffer abgelegt und enthalten Befehle welche Strings bzw. Bilder und Pictogramme an den entsprechenden Stellen im Display anzuzeigen sind, aber auch Steuerbefehle z.B. zum Setzten einer PIN sind denkbar. Die Befehle werden dann der Reihe nach abgearbeitet. Beim ersten Befehl der Displayzugriff erfordert wird zunächst der 5 V-Boost-Wandler eingeschaltet.

Das EPD-Controller-Board wird nun mit Spannung versorgt und durch die MCU über den SPI-Bus intialisiert. Dann muss das Display zunächst mit weiß überschrieben und refresht werden, anschließend wird das neue Bild durch Abarbeitung des Befehlsbuffers in den Controller geschrieben. Ist die letzte Nachricht abgearbeitet, erfolgt mit einem weiteren Refresh das Update des Displays, dann wird die Spannungsversorgung des 5 V-Wandler wieder abgeschaltet und die MCU geht wieder in den Tiefschlafmodus.



Kapitel 4

Systemimplementierung

Hier soll die Implementierung des Funktionsmusters erläutert werden. Dieses ist in Abb. 4.0.2 abgebildet. In Abb. 4.0.1 ist die Leiterplatte des Schaltungsentwurfs zu sehen. Diese wurde mit der Cadence bzw. OrCad/Allegro Toolchain entickelt. Die Entwicklung der MCU-Firmware erfolgte in der Code-Composer-Studio-IDE von Texas Instruments, die Entwicklung der Android-App in der Android-Studio-IDE.

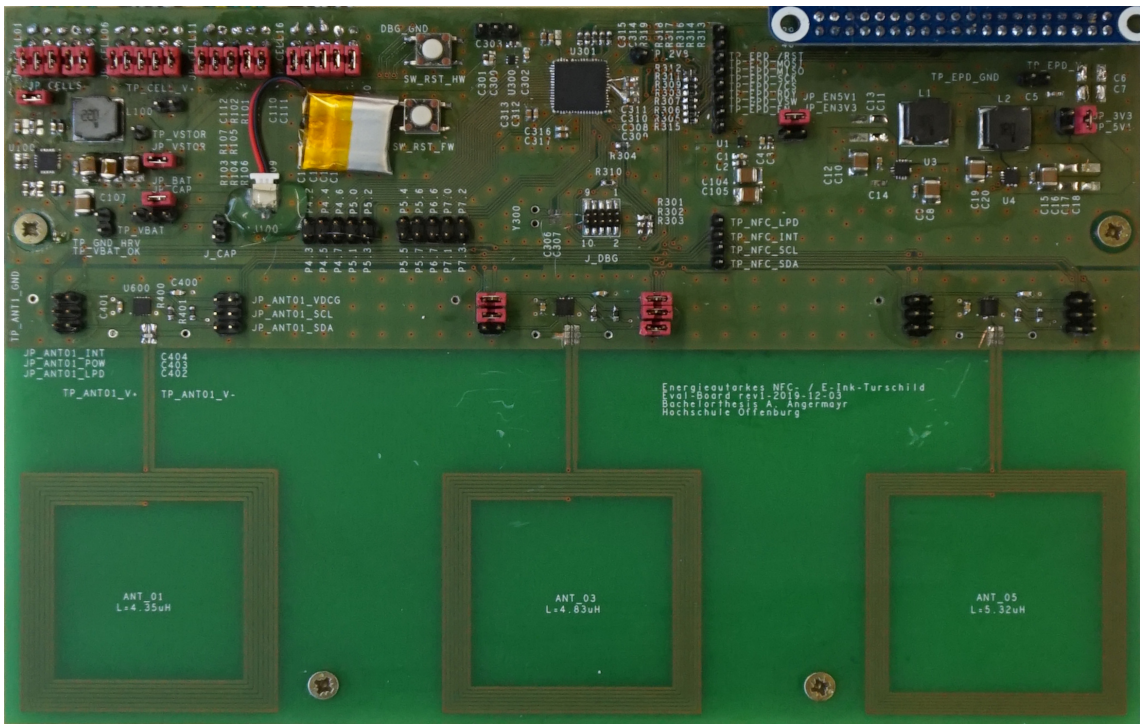


Abbildung 4.0.1: Leiterplatte des in dieser Arbeit entwickelten Funktionsmusters.

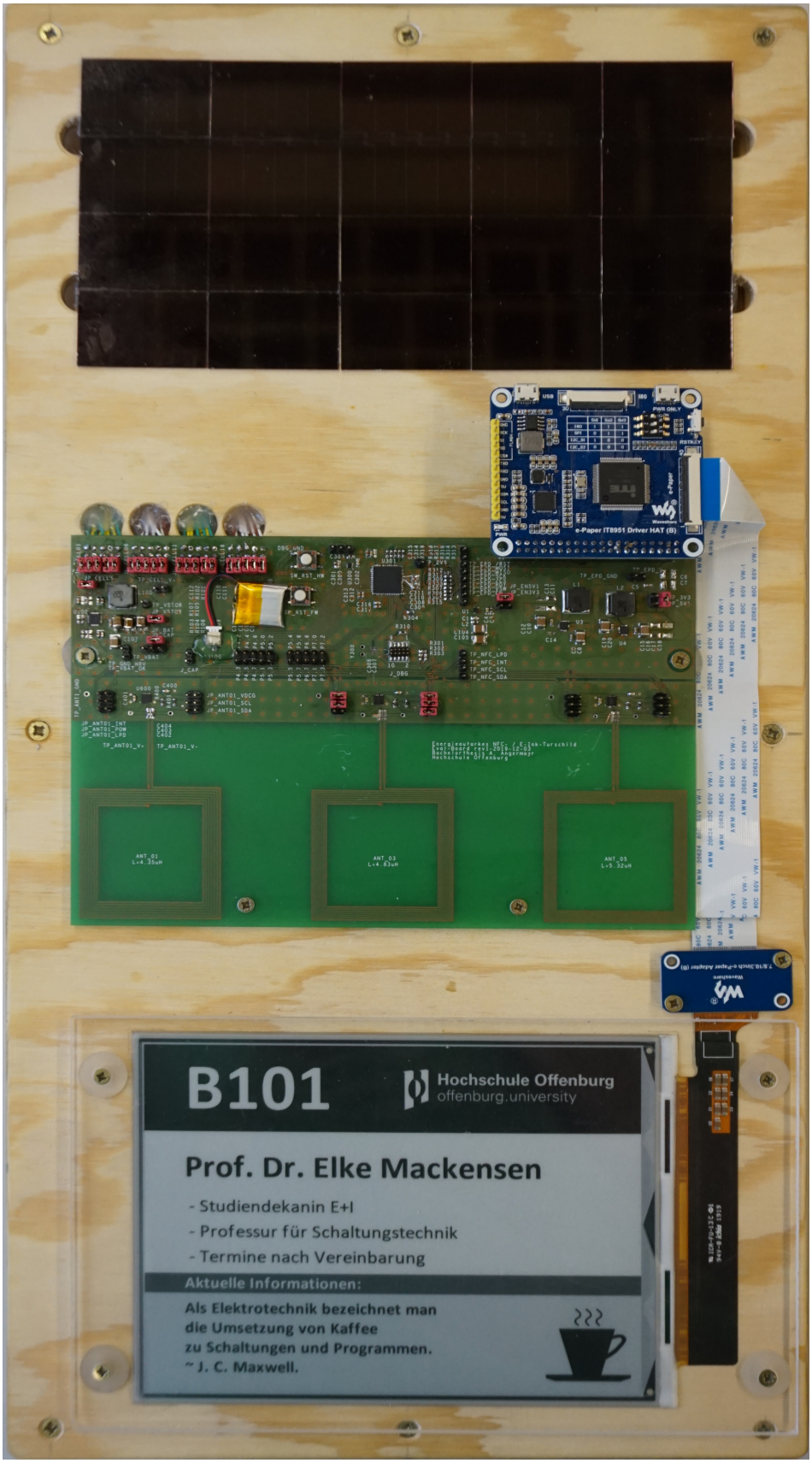


Abbildung 4.0.2: Das in dieser Arbeit entwickelte Funktionsmuster.

4.1 Implementierung der Hardware

In diesem Abschnitt soll die Implementierung des Hardwareentwurfs des Funktionsmusters mit der zugehörigen Leiterplatte erläutert werden. Diese ist als Evaluationsboard mit vielen Jumpern und Testpunkten ausgeführt und wird im Folgenden als Eval-Board bezeichnet.

4.1.1 Energy-Harvester-Teil

Zunächst wird die genaue Verschaltung des Energy-Harvester-Teiles des Eval-Boards erläutert. Dieses besteht aus den 20 Solarzellen, dem LiPo und dem `BQ25505`-Energy-Harvester/Laderegler-IC als zentralem Baustein. Abb. 4.1.1 zeigt den entsprechenden Schaltplan.

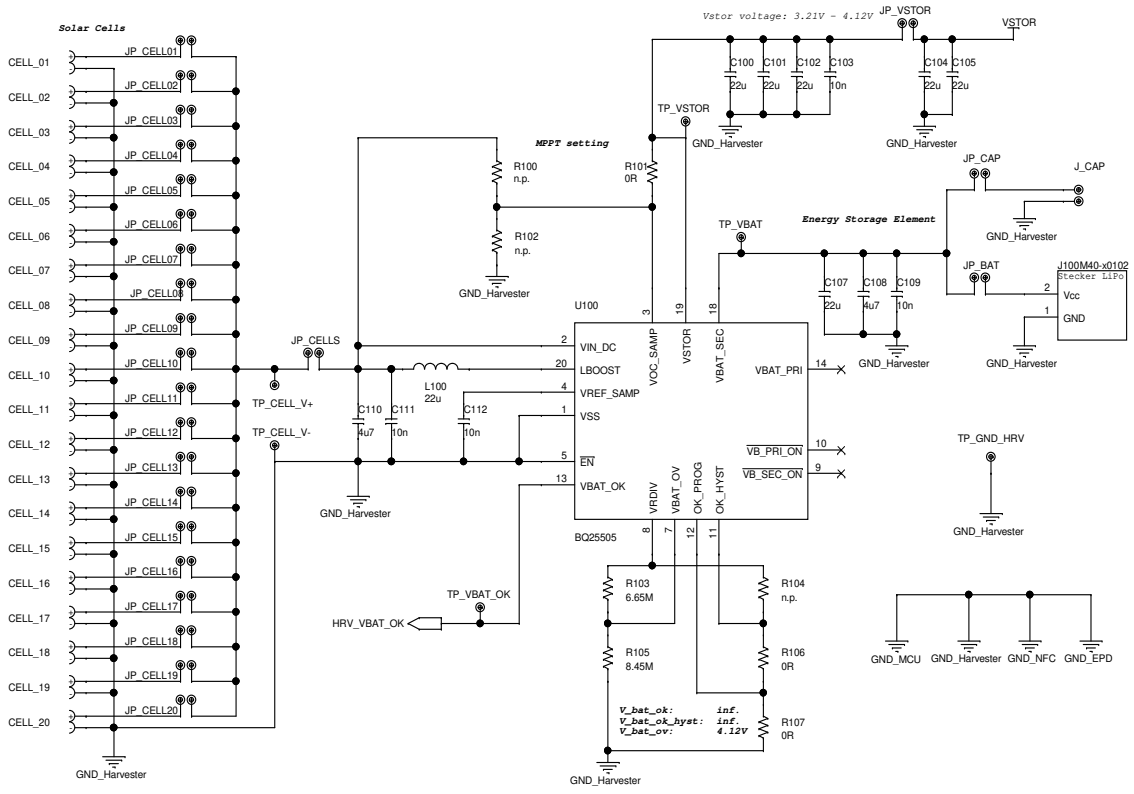


Abbildung 4.1.1: Schaltplan des Energy-Harverser-Teiles der Hardware.

Links sind die Kontakte der 20 Solarzellen zu sehen. Diese werden parallel, ohne zusätzliche Diode verschaltet. Dies ist problemlos, weil die 20 Zellen dicht beieinander platziert werden, sodass i.d.R. alle Zellen ähnlich beleuchtet werden (s. Absch. 5.3). Die Kathoden der Zellen liegen auf Masse, die Anoden der Zellen sind einzeln über die Jumper `JP_CELLxy` zuschaltbar.

Die MPP-Einstellung am Harvester-IC erfolgt über R100 - R102 und ist so bestückt, dass der MPP bei 80% der Leerlaufspannung liegt. Diese wird periodisch alle 16 s gesamplet und in C112 gespeichert.

Die Ladeschlusspannung bzw. Ziel-Spannung V_BAT_OV des internen Boost-Wandlers wird über R103 und R105 durch die Beziehung

$$\texttt{V\_BAT\_OV} = 1.815\,\mathrm{V}\left(1 + \frac{\texttt{R103}}{\texttt{R105}}\right)$$

eingestellt[2, S. 13] und ist mit einer Ungenauigkeit von etwa $\pm 0,5\%$ behaftet. Um Überladung auszuschließen, wurde $\texttt{V\_BAT\_OV} = 4,12\,\mathrm{V}$ gewählt.[1] Die Entladeschlusspannung ist nicht einzustellen, sondern ist intern auf 2 V festgelegt, d.h. der Akku ist nur durch dessen interen Schutzbeschaltung vor Tiefentladung unter $3,0\,\mathrm{V}$ geschützt.

Über R104, R106 und R107 lassen sich Schwellwert und Hysterese des VBAT_OK-Logik-Signals einstellen, da nicht benötigt, wurden die entsprechenden Pins auf Masse gelegt. Trotzdem ist das Signal HRV_VBAT_OK für eine spätere Vewendung an Pin P2.0 der MCU geführt.[2]

Der interne Boostconverter lädt bei genügend Solar-Leistung die Kapazitäten am VSTOR-Pin auf. Das System selbst wird daraus über den Jumper JP_VSTOR versorgt.

Ist die Spannung am VSTOR-Pin trotz Systemlast stabil über etwa $1,8\,\mathrm{V}$, wird der Speicher am VBAT_SEC-Pin über einen internen P-FET zugeschaltet, womit der Akku ge- bzw. entladen wird, wenn die Eingangsleistung über bzw. unter der vom System benötigten Leistung liegt, was im Ruhemodus gegeben ist.

Über die Jumper JP_CAP und JP_BAT kann ein Kondensator bzw. der LiPo als Energiespeicher zugeschaltet werden.

Die große Kapazität am VSTOR-Pin ist notwendig, weil der interne P-FET einen $R_{\mathrm{DS,on}}$ von bis zu $1,5\,\Omega$ hat. Bei zu geringer Kapazität kann bei LiPo-Spannung kleiner als $3,8\,\mathrm{V}$ durch die hohe Quellimpedanz während eines Updates nicht die benötigte Spitzenleistung geliefert werden, es kommt zu Spannungseinbrüchen und der Display-Controller hängt sich auf.

4.1.2 Mikrocontroller-Teil

Abb. 4.1.2 zeigt die Verschaltung um den Mikrocontroller. Links oben ist der NCP170 LDO-Linearregler zu sehen, der die $2,8\,\mathrm{V}$ zur Versorgung der MCU und des VDCG-

[1] Die Leiterplatte ist um die hochohmigen Widerstände R103 - R107 peinlich genau zu reinigen, ansonsten können sich Verzerrungen der Spannungsteiler ergeben.

[2] Bei Veränderung der Einstellung ist die Firmware der MCU so anzupassen, dass Pin P2.0 nichtmehr als Ausgang konfiguriert wird.

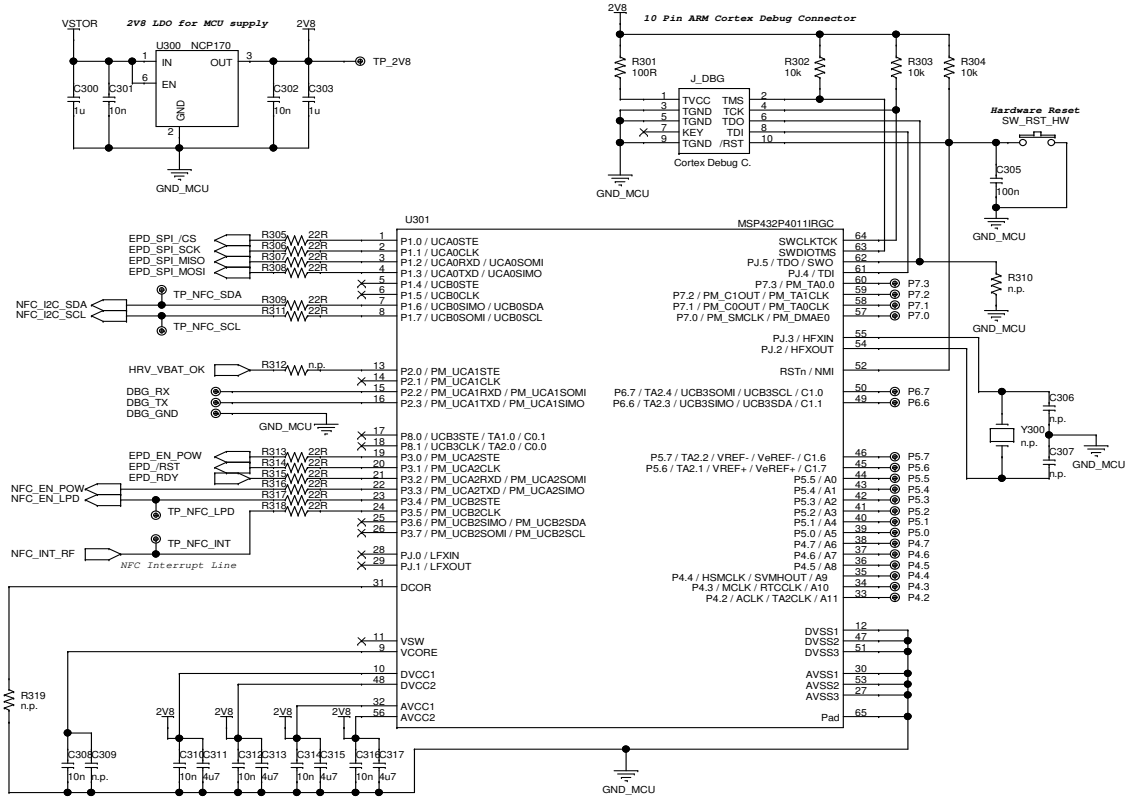


Abbildung 4.1.2: Schaltplan des Mikrocontroller-Teiles der Hardware.

Pins des NFC-Transceiver-ICs (s. Absch. 4.1.4) bereistellt. Es war notwendig einen $2,8\,\mathrm{V}$ zu wählen, damit der LDO bis zur Entladeschlusspannung des LiPo von $3,0\,\mathrm{V}$ eine saubere Spannung liefert.

Im Folgenden werden die Signalgruppen beleuchtet:

- **EPD_SPI_x** Kennzeichnet die SPI-Bus-Signale zum EPD-Controller-Board. Dieses wird zwar mit $5,1\,\mathrm{V}$ versorgt, der Controller-IC jedoch mit $3,3\,\mathrm{V}$. Entsprechend liegt die minimale `HIGH`-Spannung bei $2,31\,\mathrm{V}$[21, S. 33], sodass kein Level-Shifter für die SPI-Kommunikation mit dem Display-Controller notwendig ist. Die $22\,\Omega$-Serienwiderstände `R305 - R308` dienen zum Schutz der internen Clamping-Dioden an den MCU-Pins vor den $3,3\,\mathrm{V}$-`HIGH`-Pegeln des Controller-Boards.

- **EPD_x** Steuer-Signale vom und zum EPD-Controller-Board. `EPD_EN_POW` dient zur Aktivierung des $5,1\,\mathrm{V}$-Wandlers, `EPD_/RST` zum Reset des Controllers und `EPD_RDY` signalisiert wenn der Controller bereit ist neue Daten zu empfangen. Auch hier dienen die Serienwiderstände zum Schutz der MCU-Pins, oder können zur Unterbrechung der Leitung z.B. zur Strommesunngung verwendet werden.

- **NFC_x** Steuer- und I²C-Signale zum ST25DV-NFC-Transceiver. NFC_EN_POW an P3.3 dient zur Versorgung des Transceivers und ist im Ruhezustand ausgeschaltet (s. Absch. 4.1.4). Der Serienwiderstand R316 bildet mit den 100 nF am Tranceiver IC einen Tiefpass mit $f_g = 72\,\text{kHz}$ zur Bedämpfung von HF-Störungen.

- **DBG_x** UART-Signale zur Ausgabe von Debug-Informationen, wenn die entsprechenden #defines gesetzt sind (s. Absch. 4.2.1). Auch hier dienen Serienwiderstände zum Schutz der MCU-Pins oder zur Unterbrechung der Leitung.

- **Px.y** GPIO-Pins die auf Stiftleisten geführt wurden, um das System zu erweitern, oder Debug-Ausgaben zu tätigen (*Spion-LED*).

An Pin P2.0 könnte bei Bestückung von R312 zusätzlich das HRV_VBAT_OK-Signal ausgewertet werden. Am VCORE-Pin sind 10 nF gegen Masse für den MCU-internen LDO zu kontaktieren.

Rechts oben im Schaltplan findet sich der 10-Pin-Cortex-M-Verbinder[27, S. 2] mit 1,27 mm Rastermaß. Verbinder und MCU unterstützen Debugging über JTAG- oder SWD-Schnittstellen. Als Debugger können z.B. MSP432-Development-Boards [28] dienen, die mit einem on-Board-Debugger und ebenfalls einem 10-Pin-Debug-Verbinder im 1,27 mm-Rastermaß bestückt sind. Die Beschaltung wurde dem Schaltplan des MSP-EXP432E401Y-Development-Boards entommen[29, S. 4].

Mit SW_RST_HW ist ein Reset-Taster vorhanden, der einen Hardware-Reset am RSTn-Pin auslöst. Dieser wird durch C305 entprellt.

Die MCU wird durch den energiesparenden internen DCO-Taktgeber mit 48 MHz getaktet[25, S. 139]. Dieser ist relativ ungenau, jedoch ist UART-Kommunikation mit Baudrate 115200 problemlos möglich. Über R319 könnte der DCO zusätzlich justiert werden, weiterhin ist an den Pins HFXIN und HFXOUT eine Bestückung mit einem externen Quarz-Oszillator Y300 und zugehörigen Kapazitäten C307 und C307 möglich.

4.1.3 Display-Controller-Teil

Der Display-Controller-Teil des Eval-Boards besteht aus einem Load-Switch, über den zwei getaktete Schaltregler mit Ausgangsspannungen 5,1 V und 3,3 V versorgt werden, wovon immer nur einer zur Versorgung des Controller-Boards benötigt wird. Weiterhin ist eine 2×20-Pin-Stifleise zum Aufstecken des Controller-Boards vorhanden. Abb. 4.1.3 zeigt den zugehörigen Schaltplan.

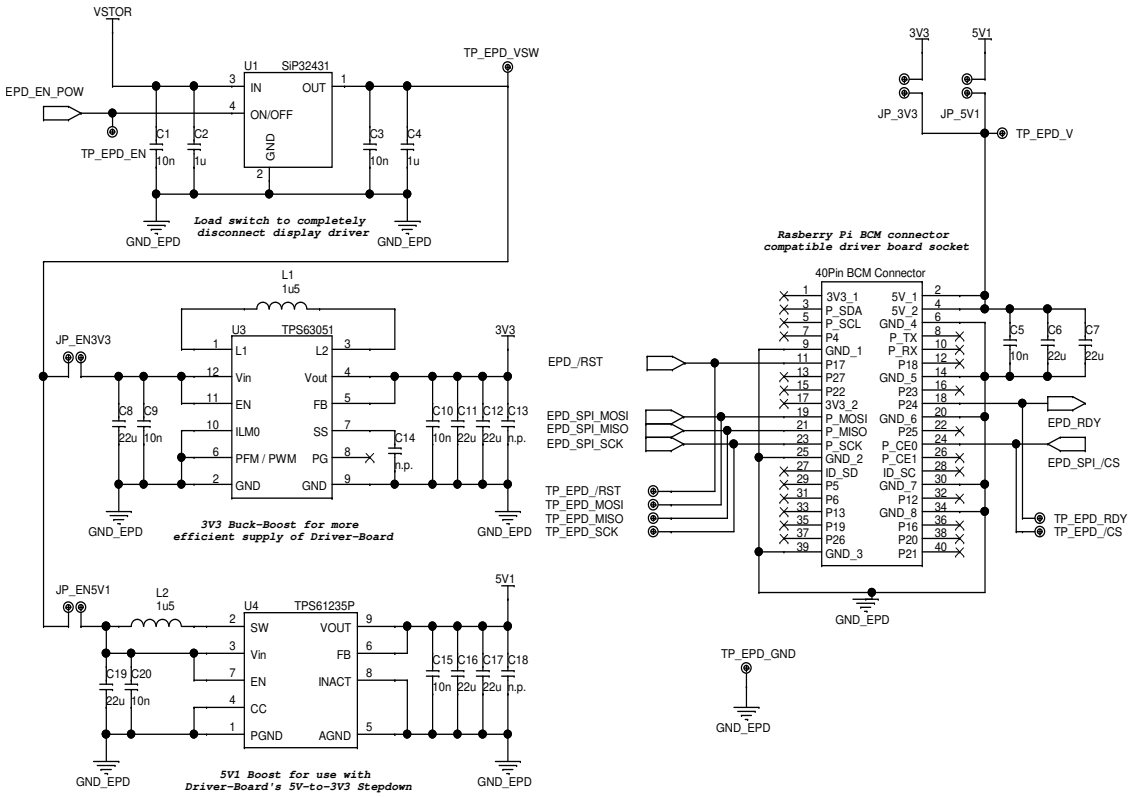


Abbildung 4.1.3: Schaltplan des Display-Controller-Teiles der Hardware.

Laut Hersteller benötigt das Controller-Board zwar eine 5 V Versorgungsspannung, jedoch ist auf dem Controller-Board zusätzlich ein 3,3 V-Buck-Wandler verbaut, der den `IT8951`-Controller versorgt[30]. Weiterhin ist auf dem Board ein `TPS651851`-Power-Management-IC (PMIC) vorhanden, ein spezieller PMIC mit mehreren Schalt- und Linearreglern zum Treiben von E-Paper-Displays[31]. Dieser wird auf dem Board mit 5 V versorgt, toleriert aber Eingangsspannungen zwischen 3 V und 6 V. Idee war nun, den 3,3 V-Wandler auf dem Controller-Board zu deaktiviern und zu überbrücken und das gesamte Board mit 3,3 V zu versorgen, um nicht erst die LiPo-Spannung verlustbehaftet auf 5 V zu wandeln, die der Wandler auf dem Controller-Board dann wieder verlustbehaftet auf 3,3 V herabsetzt. Hierzu ist dann ein modifiziertes Controllerboard nötig.

Über die Jumper `JP_EN5V1` und `JP_EN3V3` kann der gewünschte Wandler ausgewählt werden, entsprechend müssen die Jumper am Ausgang `JP_5V1` und `JP_3V3` gesteckt[3] werden.

Der 5,1 V-Wandler ist ein `TPS61235P` von Texas Instruments und ein reiner Boost-Wandler mit hohem Wirkungsgrad bis 97%. Die Schaltfrequenz beträt 1 MHz

[3]Es darf immer nur **ein** Jumper `JP_5V1` und `JP_3V3` gesteckt sein!

und liegt damit unter der NFC-Frequenz von 13,56 MHz[4] , der maximale Drosselstrom liegt bei etwa 8 A[32, S. 5].

Als 3,3 V-Wandler ist ein Buck-Boost-Wandler notwendig, weil die Akkuspannung zwischen 3 V und 4,12 V liegen kann. Es wurde ein `TPS63051` von Texas Intruments verwendet, mit einem maximalen Drosselstrom bis 2 A[33, S. 6] und Wirkungsgraden bis etwa 95%. Geschaltet wird mit 2,5 MHz.

Die Kapazitäten `C10` - `C13` sowie `C15` - `C18` an den Ausgängen der Wandler sind gegebenenfalls noch größer zu wählen, um die vom EPD benötigten Spitzenleistungen liefern zu können. Als Speicherdrosseln kommen zwei verlustarme geschirmte 8 mm × 8 mm-Drosseln[34] mit 1,5 µH Indutivität bei 10 mΩ DC-Widerstand und einem Nennstrom von 5,65 A zum Einsatz.

Als Loadswitch wurde ein sparsamer `SiP32431` von Vishay verwendet. Dieser verkraftet Strompulse bis 3 A, die Leckströme summieren sich je nach Eingangsspannung lediglich zu einigen hundert Nanoampere. Zu bemerken ist, dass der $R_{\mathrm{DS,On}}$ mit bis zu 290 mΩ die Quellimpedanz der Controller-Board-Versorgung nochmals verschlechtert (zusätzlich zu den 1,5 Ω des `BQ25505`, s. Absch. 4.1.1).

Rechts unten im Schaltplan findet sich schließlich die Doppelstiftleiste zum Aufstecken des Controller-Boards. Die einzelnen Signale wurden bereits in Absch. 4.1.2 beleuchtet, für Informationen zu deren Weiterverschaltung auf dem Controller-Board sei auf dessen Schaltplan[30] verwiesen.

4.1.4 NFC-Transceiver-Teil

Der NFC-Transceiver-Teil ist in dreifacher Ausführung auf dem Eval-Board vorhanden, unterschiedlich sind jeweils lediglich die Induktiviätswerte L_{Ant} der Antennenspulen. Diese sind wie für NFC üblich direkt als Leiterbahnschleifen auf der Leiterplatte ausgeführt. Die errechnete Induktivität weicht dabei in der Regel von der tatsächlichen ab, daher die dreifache Ausführung.

Entwurf der NFC-Antenne

Ziel ist es, dass die Antennenspule mit parallel zu ihr geschalteten (parasitären) Kapazitäten einen Parallelschwingkreis mit Resonanzfrequenz $f_{\mathrm{r}} = 13,56\,\mathrm{MHz}$ bilden, bestehend aus L_{Ant}, der Parallel-Kapazität C_{Ant} und Parallel-Widerstand R_{Ant}[5].

[4] Prinzipiell ist die Schaltfrequenz im vorliegenden System unkritisch, da NFC-Kommunikation immer *vor* dem Einschalten der Schaltwandler stattfindet.

[5] R_{Ant} als *Parallel*-Widerstand der Antenne ist nur indirekt-reziprok über den *Serien*-Widerstand R_{S} der Antenne (der Widerstand der Spulenwindungen) beeinflussbar, d.h. mit steigendem R_{S} sinkt R_{Ant}.

Der Gütefaktor Q dieses Schwingkreises ist gegeben[35, S. 306] durch

$$Q = \frac{I_\mathrm{L}}{I} = \frac{R_\mathrm{Ant}}{\omega_\mathrm{r} L}, \tag{4.1}$$

wobei I den Gesamtstrom durch den Schwingkreis und I_L den Teilstrom durch die Antennenspule kennzeichnet. Die Bandbreite B ist dabei mit der Güte über

$$B = \frac{f_\mathrm{r}}{Q}, \quad \text{mit} \quad f_\mathrm{r} = \frac{1}{2\pi\sqrt{L_\mathrm{Ant} C_\mathrm{Ant}}} \tag{4.2}$$

verknüpft. Weiterhin gilt für die Spannung u_L über der Antennenspule[36, S. 252]

$$u_\mathrm{L} = L_\mathrm{Ant} \frac{\mathrm{d}i_\mathrm{L}}{\mathrm{d}t}. \tag{4.3}$$

Vom Transceiver detekiert wird die Spannung u_L, d.h. diese ist zu maximieren, womit nach (4.3) L_Ant zu maximieren ist. Da C_Ant in (4.2) durch die (parasitäre) Kapazität C_tun zwischen den Antennen-Pins ($28,5\,\mathrm{pF} \pm 2\,\mathrm{pF}$ beim `ST25DV`) nach unten begrenzt ist, ergibt sich daraus eine Obergrenze für L_Ant mit

$$L_\mathrm{Ant} \leq \frac{1}{(2\pi f_\mathrm{r})^2 \cdot C_\mathrm{Ant}}.$$

Die Güte des Schwingkreises beeinflusst durch die Leistungsdissipation im Parallelwiderstand die Reichweite und Menge der über das Feld übertragbaren Energie und ist im vorliegenden System zweitranging, da kein Energy-Harvesting aus dem NFC-Feld erfolgen soll.

Gemäß (4.2) sinkt mit der Güte die Bandbreite. Bei zu schmaler Bandbreite wird die Trägerschwingung nicht ausreichend bedämpft sodass die Symbol-Abfallszeiten zu groß werden können und ein zu langes Nachoszillieren des Schwingkreises zu Symbolsynchronisationsfehlern führen kann[37, S. 24]. Weiterhin müssen die Trägerfrequenz des Readers und die Resonanzfrequenz des Tags bei zu kleiner Güte sehr nahe beieinander liegen, da der Träger ansonsten zu stark bedämpft wird.

Daraus folgt, dass eine kleine Güte für die Kommunikation vorteilhafter ist, sich aber negativ auf die Energieübertragung auswirkt.[38, S.20ff]. Als Kompromiss wird in der Praxis daher eine Güte um $Q \leq 20$ gewählt[39, S.13] und entsprechend eine Bandbreite von 680 kHz

Zur Auslegung der Antenne wurde der Prozedur aus [40] unter Verwendung des *eDesign-Antenna*-Werkzeugs von ST[41] gefolgt. Letzteres ist eine browserbasierte Applikation zur Dimensionierung von Leiterplatten-Antennen für NFC. Nach Eingabe der gewünschten Parameter der Antenne wird daraus die Induktivität bei

13.56 MHz berechnet. Zunächst wurden nun die Abmaße festgelegt, dann itterativ die Breite und Anzahl der Windungen verändert, bis sich eine gewünschte Ziel-Induktivität von $L^*_{\mathrm{Ant}} = 4.83\,\mathrm{MHz}$ ergab. Da zu erwarten war, dass dieser Wert nur näherungsweise mit der tatsächlichen Induktivität übereinstimmt, wurden duch Vergrößerung bzw. Verkleinerung der Leiterbahnbreite wie in Tab. 4.1 gezeigt zwei[6] weitere Antennen entworfen.

Tabelle 4.1: Entwurf der 3 Antennenspulen.

Parameter	**ANT_01**	**ANT_03**	**ANT_05**
L_{Ant}:	4,35 µH	4,83 µH	5,32 µH
% von L^*_{Ant}:	-10%	0%	+10%
Abmaße L×B:	45 mm×45 mm	45 mm×45 mm	45 mm×45 mm
Windungen:	7	7	7
Abstand Windung-Windung:	0,2 mm	0,2 mm	0,2 mm
Serienwiderstand:	0,9 Ω	1,2 Ω	1,6 Ω

Die induktive Kopplung zwischen Reader und Tag ist maximal, wenn die Antenneninduktivitäten genau deckungsgleich sind, d.h. auch die gleichen Abmaße haben. Die Größe der Antenne basiert auf Erfahrungswerten bezüglich der Größe von NFC-Antennen einige Smartphones. Zur Vermessung der Antennen war es notwendig auch den Transceiver dreifach auf dem Eval-Board auszufühhren[7].

Abgleich und Vermessung der Antennen

Ziel des Abgleichs der Antenne ist es, die Resonanzfrequenz auf $f_{\mathrm{r}} = 13,56\,\mathrm{MHz}$ zu stimmen. Hierbei stellt sich die Frage, wie am Schwingkreis Spannung oder Strom gemessen werden kann, denn jede Kontaktierung mit einem Kabel verzerrt den Schwingkreis. Eine elegante Methode wurde von Hr. Ing. Moser in [42] gefunden: Bei Resonanz ist die Spannung U_{EH} am Energy-Harvesting-Pin des `ST25DV` maximal, daher kann der Abgleich durch Maximierung von U_{EH} erfolgen. Der Schwingkreis wird dabei nicht verzerrt, weil keine zusätzlichen Bauteile bzw. Leitungen an diesen kontaktiert werden müssen, die IC-interne Energy-Harvester-Schaltung schützt den Schwingkreis vor den parasitären Effekten der Messleitungen.

Es wurde nun eine große Antenne, die zwischen 10 MHz und 20 MHz Anregungsfrequenz einen quasilinearen Frequenzgang hat, mittels Funktionsgenerator bei ver-

[6] Die Nummerierung der Antennen kommt daher, dass ursprünglich 5 Antennen entworfen wurden, mit Toleranzen von ±10%, ±5% um L^*_{Ant}.

[7] Ein Wechsel der Antennen z.B. durch Jumper kam nicht infrage, da diese aufgrund parasitäter Eigenschaften den Schwingkreis verzerren würden, womit bei einem Folgeprojekt, wo nur noch eine Antenne ohne Jumper verwendet würde, die hier gemachten Messergebnisse bezüglich Antenneninduktivität und Resonanzfrequenz wertlos wären.

schiedenen Frequenzen f mit Spitzen-Spitzen-Amplitude $A_{\mathrm{PP}} = 10\,\mathrm{V}$ gespeist und unter die Antennen des Eval-Boards platziert. Mittels Multimeter wurde die Spannung am Energy-Harvesting-Pin gemessen.

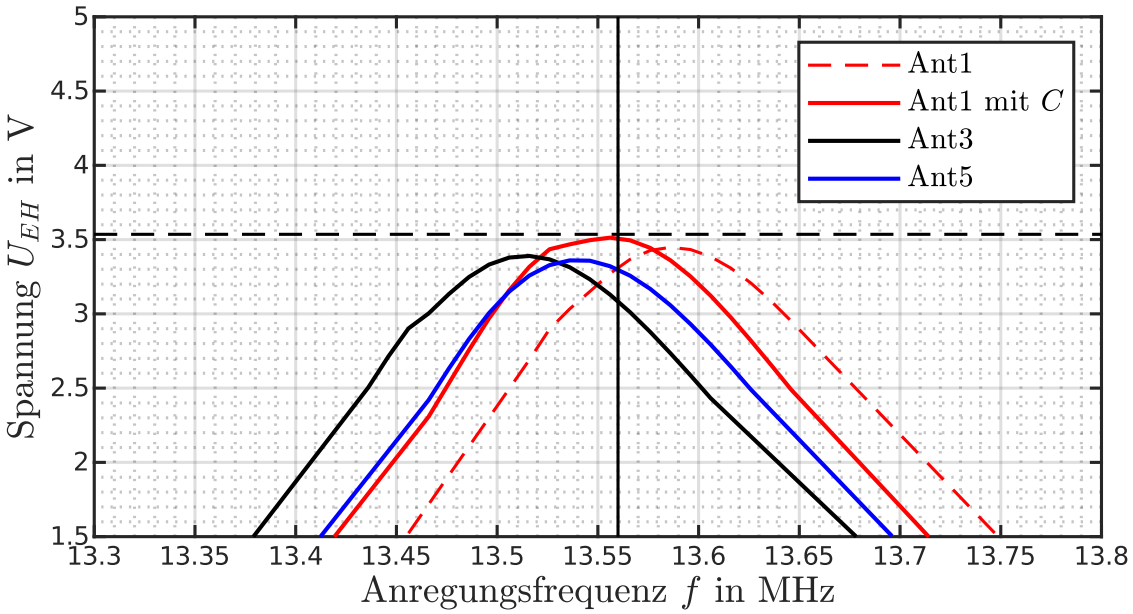


Abbildung 4.1.4: Spannung am Energy-Harvesting Pin des ST25DV über der Anregungsfrequenz für die 3 Antennen.

Abb. 4.1.4 zeigt die Messergebnisse. Die Resonanzfrequenzen von Antennen ANT_03 und ANT_05 lagen knapp unter $13,56\,\mathrm{MHz}$, hier ist keine Anpassung durch Erhöhung der parallelen Kapazität mehr möglich. Die Resonanzfrequenz von ANT_01 lag knapp über $13,56\,\mathrm{MHz}$, sodass eine Anpassung durch Einlöten eines NP0-Keramik-Kondensators mit $C = 10\,\mathrm{pF}$ erfolgen konnte.

Die Spitzenspannungen liegen um $3,35\,\mathrm{V}$ - $3,5\,\mathrm{V}$, entsprechend betragen die $-3\,\mathrm{dB}$-Bandbreiten der Antennen bei den Amplituden $V_{\mathrm{EH,max}}/\sqrt{2}$ zwischen $1,74$ MHz und $1,71\,\mathrm{MHz}$, was Güten zwischen $7,8$ und $7,9$ entspricht.

Fazit Antennenentwicklung

Die realisierten Werte liegen insgesamt sehr nahe an den durch das eDesign-Antenna-Applikation berechneten Werten. Augenscheinlich sind alle Antennen bei einem Abstand von $30\,\mathrm{mm}$ zum Smartphone gleich performant, auch die Anpassung von Antenne 1 brachte keine merkbare Veränderung. Somit wären alle 3 Antennen trotz einer Streuung von $\pm 10\%$ des Induktivitätswertes ohne zusätzliche Anpassung nutzbar, was sich im industriellen Umfeld positiv auf die Produktionskosten auswirkt.

Es ist damit festzuhalten, dass die Auslegung und Anpassung der NFC-Antenne relativ unkritisch ist, wenn NFC nur zur Kommunikation, nicht aber zum Energy-Harvesting verwendet werden soll.

Beschaltung des NFC-Transceivers

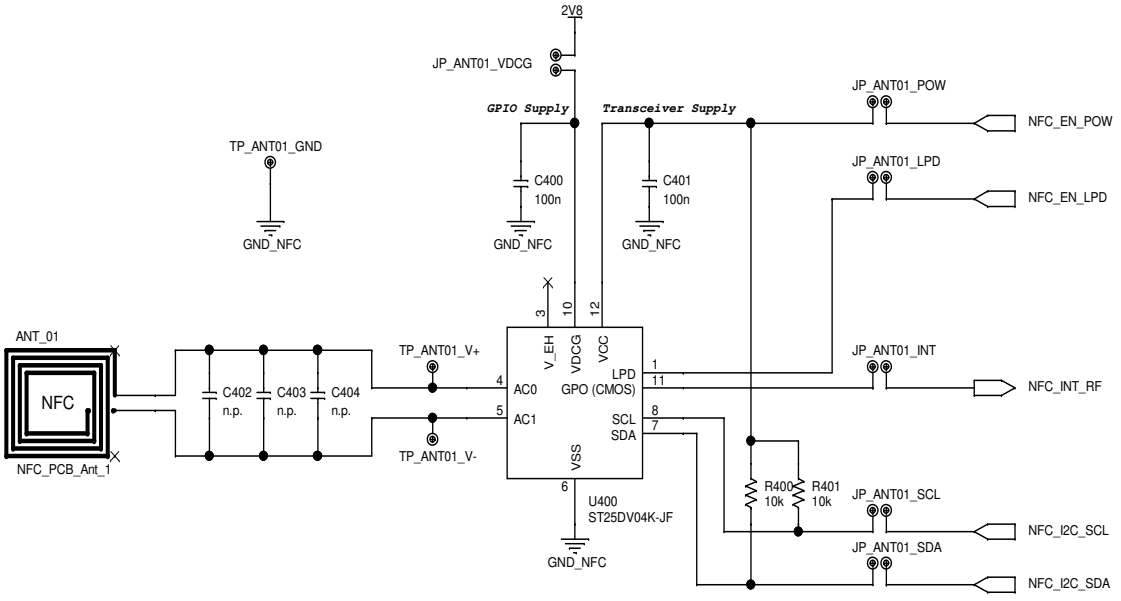


Abbildung 4.1.5: Schaltplan des NFC-Transceiver-Teiles der Hardware.

Abb. 4.1.5 zeigt den NFC-Teil des Systems. Wie oben erläutert ist diese Schaltung in 3-facher Ausführung mit unterschiedlichen Antennen vorhanden, die sich über die nach (von) rechts gehenden (kommenden) Signale über die Jumper `JP_ANTx_y` aktivieren bzw. deaktivieren lassen[8].

Zentral im Schaltplan ist der `ST25DV`-NFC-Transceiver-IC. Links ist die Antenneninduktivität zu sehen, die direkt auf der Leiterplatte realisiert wird. Die Bestückungsplätze `C402` - `C404` in Bauteilgröße `0603` können mit Kondensatoren zum Abgleich oder mit Widerständen zur Erhöhung der Bandbreite bestückt werden.

[8]Es sollte zeitgleich immer nur ein Transceiver benutzt werden.

Der `V_EH`-Pin des Transceivers wurde nicht kontaktiert, da dieser nur für den Abgleich benötigt wird[9]. Der `VDCG`-Pin dient lediglich zur Versorgung der Ausgangsstufe des Interrupt-Pins (s. Absch. 3.5.1) und hat laut Datenblatt einen Leckstrom von höchstens 100 nA bei 5, 5 V Versorgung.

Versorgt wird der Transceiver bei Bedarf direkt über den MCU-Pin am `NFC_EN_POW`-Signal, über das `NFC_EN_LPD`-Signal ließe sich der Transceiver in einen Energiesparmodus versetzen, das Abschalten der Versorgunsspannung am `VDC`-Pin ist aber noch stromsparender.

Am `GPO`-Pin wird bei NFC-Aktivität ein Interrupt an die MCU ausgegeben, die `NFC_I2C_x`-Signale bilden Daten- und Taktsignal für den I^2C-Bus zur MCU. Die hierfür notwendigen Pullup-Widerstände werden abschaltbar aus der Versorgung des Transceivers versorgt, prinzipiell würden sich aber auch die MCU-internen Pullup-Widerstände an den Pins verwenden lassen.

4.2 Implementierung der Firmware

In diesem Abschnitt soll der Aufbau und die Implementierung der Firmware, d.h. der MCU-Software, vorgestellt werden. Insbesondere wird erläutert, wie die Firmware die im vorherigen Abschnitt eräuterten Systemfunktionen bereitstellt und welche Energiesparmaßnahmen getroffen werden.

Dazu wird zunächst ein Überblick über die Softwarearchitektur sowie das SKEID-Kommunikationsprotokoll gegeben. Letzteres spezifiziert Synchronisation, Syntax und Semantik der Kommunikation zwischen Android-App und Türschild, sowie den Satz der Befehle zur Steuerung des Türschildes. Dann werden einzelne Module näher untersucht, abschließend folgt die Vorstellung des Programmablaufs der Main-Funktion und der zugehörigen NFC-GPO-Pin-Interrupt-Service-Routine (s. Absch. 3.2.1).

4.2.1 Softwarearchitektur

Abb. 4.2.1 zeigt den hierarchischen Aufbau der Module die zusammen die Firmware bilden. Die einzelnen Module sind in blau, die zugehörige Ordner-Hierarchie in grau dargestellt. Zuoberst ist die `main.c`-Anwendung, sowie das `irq_handler`-Modul, welches die NFC-GPO-Interrupt-Signale verarbeitet. Beide greifen auf die Module im *nfc*-Ordner zu. Das `tal.h`-Modul (Transmitter-Abstraction-Layer)-Modul abstrahiert den NFC-Transceiver, der Methoden für die Abwicklung der Kommunikation

[9] Es wäre hierfür sehr praktisch gewesen diesen als Testpunkt zu kontaktieren.

mit der Smartphone-App nach dem SKEID-Protokoll bereitstellt und speichert die empfangenen Befehle im Nachrichten-Buffer `ringbuffer.h`. Dazu verwendet es Methoden der Module im *st25dv*-Ordner, die aus dem `ST25DV`-Software-Development-Kit[43] stammen. Diese wiederum verwenden die Treiber im *i2c*-Ordner für die Kommunikation mit dem NFC-Transceiver.

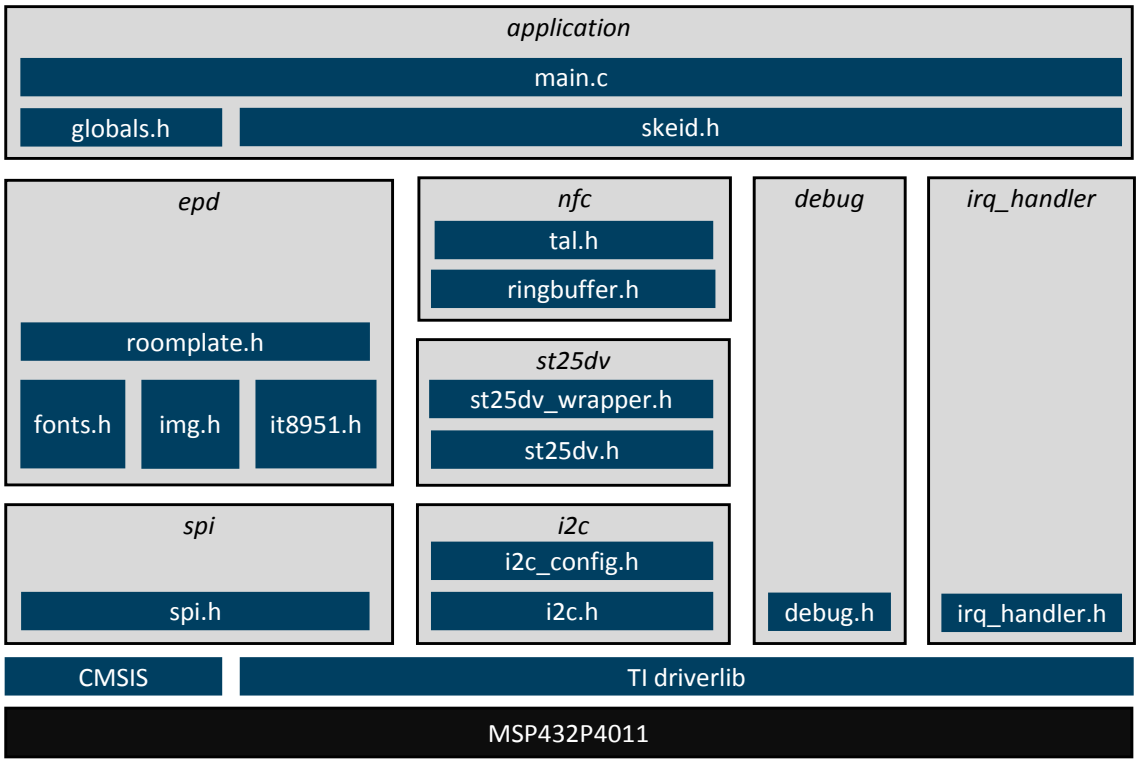


Abbildung 4.2.1: Softwarearchitektur der MCU-Firmware. *Kursive* Modulgruppen entsprechen Ordner-Hierachie.

Nach Vollendung der Kommunikation werden in der `main.c`-Anwendung die Befehle aus dem `ringbuffer.h` ausgelesen, und die zugehörigen Methoden im `tal.h`-Modul aufgerufen. Diese verarbeiten die Befehle und greifen dabei insbesondere auf das `roomplate.h`-Modul zu, welches das Türschild-Display abstrahiert und den Pixel-Buffer des EPD-Controllers beschreibt. Hierfür werden Methoden des Display-Controller-Treibers im `it8951.h`-Modul verwendet, die Module `fonts.h` und `img.h` stellen die Bitmap-Arrays mit Schriften und Piktogrammen bzw. Bildern bereit. Weiterhin werden Methoden des SPI-Treibers im *spi*-Ordner verwendet. Das `debug.h`-Modul schließlich stellt per `#defines` hierarchisch aktivierbare Systemausgaben über die serielle Schnittstelle an der `DBG_x`-Stifleiste (s. Absch. 4.1.2) bereit, die zugehörigen Treiber sind im selben Modul implementiert.

4.2.2 SKEID-Kommunikationsprotokoll

Hier soll ein kurzer Überblick zum SKEID-Kommunikationsprotokoll gegeben werden. Dieses ist rahmenweise organisiert, wobei jeder Rahmen einen Befehl sowie Steuerzeichen und gegebenenfalls Daten enthält. Daher soll zunächst das Rahmen- bzw. Übertragungsformat untersucht werden, anschließend wird der Befehlssatz vorgestellt.

Rahmenformat Abb. 4.2.2 zeigt die Rahmenformate und den zeitlichen Ablauf der Kommunikation an einer beispielhaften Übertragung.

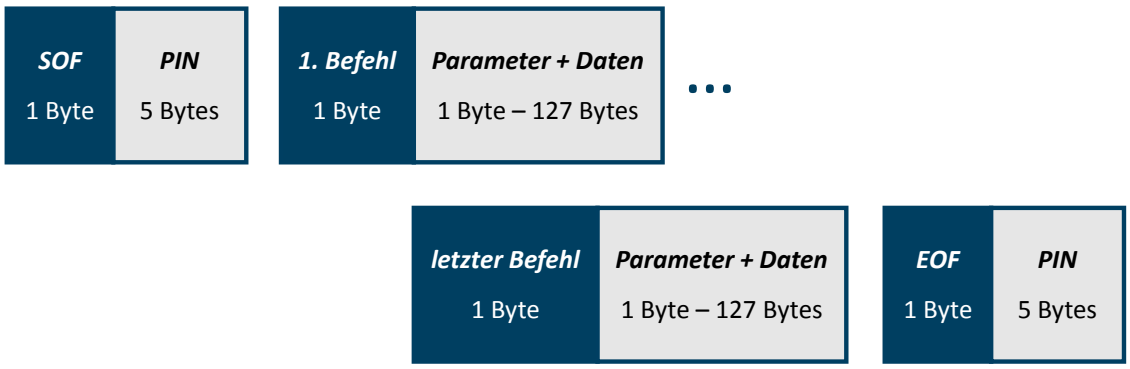


Abbildung 4.2.2: Zeitlicher Ablauf des SKEID-Kommunikations-Protokolls.

Grundsätzlich enhält jeder Rahmen einen 1 B-Befehlscode, sowie bis zu 127 B an Daten[10]. Am Beginn jeder Kommunikation zwischen App und Türschild steht ein Start-Of-Transmission-Rahmen (SOT), gefolgt von einer fünfstelligen PIN-Nummer, zur Kennzeichnung des Beginns einer Übertragung. Die PIN dient zur Authentifizierung des Benutzers und kann grundsätzlich via App verändert werden. Entgegen Anforderung 3c wird die PIN derzeit noch unverschlüsselt übertragen, ebenso ist die PIN-Änderungsfunktion noch nicht implementiert.

Zwischen den einzelnen Rahmen darf bis zu 1 s Zeit liegen und es können bis zu 254 Befehls-Rahmen gesendet werden. Am Ende der Übertragung muss ein End-Of-Transmission-Rahmen (EOT) folgen, wieder mit der PIN, um die Übertragung als valide zu bewerten, ansonsten werden alle empfangenen Befehle verworfen. Ebenso führt das Empfangen eines erneuten SOT-Rahmens zum Löschen des Nachrichtenbuffers. Erst nach Vollendung der Übertragung werden die Befehle ausgelesen und ausgeführt.

[10] Obwohl die Mailbox des NFC-Transceivers mit 256 B deutlich größer ist, wurde 128 B als Maximallänge festgelegt, da nicht alle Smartphones in der Lage waren, 256 B lange Nachrichten zu verschicken. Nachfrage bei ST, dem Anbieter der Android-SDK, bestätigte dieses Verhalten. [44]

Befehlssatz Das erste Byte eines Rahmens enthält den Befehlscode sowie ein Steuer-Bit, wie in Tab. 4.2 gezeigt.

Tabelle 4.2: Bit-Belegung im Befehls-Byte des SKEID-Protokolls.

Bits:	**7 5**	**4 3**	**2**	**1 0**
Verwendung:	Befehls-Code		Kein-Refresh-Bit	Reseviert

Das *Kein-Refresh-Bit* dient dazu dem Türschild mitzuteilen, ob nach einem Befehl, bei dem auf den Pixel-Buffer des Display-Controllers zugegriffen wird, ein Refresh des Displays erfolgen soll. Damit sollte dieses Bit bis auf den letzten Display-Befehl immer gesetzt sein, erst nach dem letzten Befehl soll der Refresh durchgeführt werden.

In Tab. 4.3 ist der derzeitige Befehlssatz mit den zugehörigen Befehls-Codes und Funktionen aufgelistet. Dieser ist prinzipiell einfach erweiterbar.

Tabelle 4.3: Befehlssatz des SKEID-Protokolls.

Befehl	**Code**	**Funktion**
PRESENT_SOT	01000_2	Start-Of-Transmission-Rahmen zur Initialisierung des Receivers.
PRESENT_EOT	01001_2	End-Of-Transmission-Rahmen zum Abschluss und zur Validierung der Übertragung.
CHANGE_PIN	00001_2	Änderung der PIN. Enthält neue PIN als Daten Payload.
EPD_WRITE_STRING	00010_2	Überschreiben einer String-Zeile. Daten enthalten die Position als Bereichs- und Zeilenindizes, die Länge des Strings, sowie den String.
EPD_WRITE_PICT	00011_2	Überschreiben des Piktogramms. Daten enthalten den Piktogrammindex.

4.2.3 Modulbeschreibungen

Grundsätzlich ist zu sagen, dass ein Großteil des Codes der SKEID-Gruppe wiederverwendet werden konnte. Tab. 4.4 zeigt die notwendigen bzw. vorgenommenen Änderungen, dann folgen Beschreibungen einzelner Module.

tal.h Der Transmitter-Abstraction-Layer abstrahiert den Receiver und den Befehlsprozessor des SKEID-Protokolls. Hervorzuheben sind die Methoden `NFC_vCommand_IT()` und `NFC_ucExecuteCommand()`, deren Programmflussdiagramme in Abb. 4.2.3 dargestellt sind.

Tabelle 4.4: Notwendige Änderungen an der SKEID-Software.

Anpassungen	**Bermerkungen**
Main-Applikation:	Vollständige Neuentwicklung, Energiesparfunktionen implementiert
SPI-Treiber:	Anpassung Pinzugriffe, Optimierung Timing
I^2C-Treiber:	Anpassung Pinzugriffe
Display-Treiber:	Tausch des Treibers für den neuen EPD-Controller
Interrupt-Service-Routinen:	Anpassung auf die neue Mikrocontroller Architektur
Pinzugriffe:	Anpassung auf den neuen Mikrocontroller
Türschild-Layer:	Neuentwicklung zur Abstraktion des Türschild-Displays
Debug-Schnittstelle:	Neuentwicklung

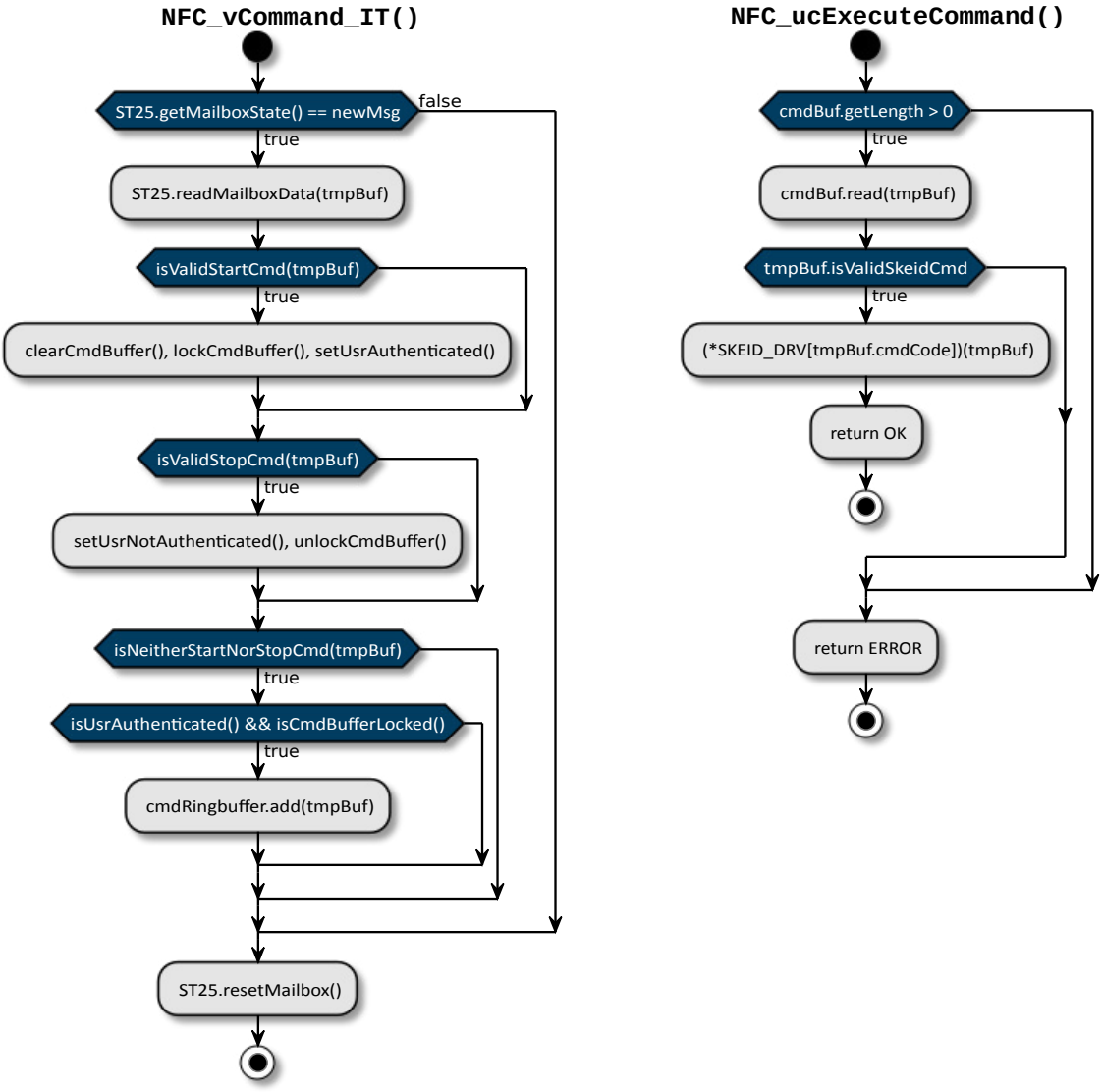


Abbildung 4.2.3: Programmflussdiagramme der `NFC_vCommand_IT()`- und `NFC_ucExecuteCommand()`-Funktionen.

- `NFC_vCommand_IT()`: Wird von der Interrupt-Service-Routine bei einem NFC-GPO-Interrupt im Wachzustand aufgerufen und steuert den Zustand des Tran-

sceivers und wickelt die Kommunikationmit dem `ST25DV` ab. Dazu wird, wie links in Abb. 4.2.3 dargestellt, zunächst überprüft, ob die Interruptursache das Eintreffen einer neuen Nachricht in der Mailbox war. Falls ja, wird diese gelesen und geprüft, ob es sich um gültige SOT- bzw. EOT-Rahmen handelt.

Bei einem gültigen SOT-Rahmen wird der Befehlsbuffer, in dem zunächst alle Befehle abgelegt werden geleert und gesperrt, sodass voererst keine Befehle aus dem Buffer durch das Main-Programm abgearbeitet werden, dann wird der Benutzer, aufgrund gültiger PIN-Eingabe, als authentifiziert betrachtet.

Bei einem gültigen EOT-Rahmen wird zunächst die Benutzerauthentifizierung, dann die Sperrung des Befehls-Speichers aufgehoben, da die Übertragung nun vollständig ist. D.h. nach Rückkehr ins Main-Programm kann dort nun der Buffer abgearbeitet werden.

Befehlsrahmen die weder vom SOT- noch vom EOT-Typ sind, werden, gegeben den Fall dass der Benutzer authentifiziert und der Befehlsbuffer gesperrt ist, im Befehls-Speicher abgelegt.

Anschließend wird die Mailbox zurückgesetzt, was unbeding notwendigt ist, da erst danach neue Nachrichten von der App in die Mailbox geschrieben werden können.

- `NFC_ucExecuteCommand()`: Wird zur Abarbeitung der Befehle im Befehls-Speicher vom Main-Programm aufgerufen, wenn der Befehls-Speicher nicht gesperrt ist. Wie rechts in Abb. 4.2.3 dargestellt, wird hierzu zunächst geprüft ob der Speicher bereits leer ist, falls nein, wird ein Befehl nach dem *first-in-first-out*-Prinzip aus dem Speicher gelesen und in `tmpBuf` abgelegt.

  Dann wird überprüft ob der Befehls-Code einen gültigen Wert hat. Falls ja, wird dieser als Index in ein Funktionszeiger-Array `SKEID_DRV[]` gegeben, wodurch der Befehls-Code effizient auf eine Funktion verweist, die den Befehl letztlich implementiert.

  Als Parameter wird der Funktion ein Zeiger auf den Datenbereich des aktuellen Rahmens im `tmpBuf` übergeben, sodass in den `SKEID_DRV`-Funktionen die empfangenen Daten zur Verfügung stehen.

roomplate.h Das `roomplate.h`-Modul abstrahiert das Türschild-Display. Dieses ist in feste, modifizierbare Bereiche bzw. Zeilen eingeteilt, wie in Abb. 4.2.4 dargestellt.

Abbildung 4.2.4: Bereiche und Zeilen auf dem Türschilddisplay.

In Tab. 4.5 sind die Eigenschaften der Text-Bereiche aufgelistet. Im Feld *Bild-Index* lässt sich zusätzlich ein Piktogramm der Größe 320 px × 320 px aus dem `pictograms[]`-Bitmap-Array im `img.h`-Modul darstellen.

Tabelle 4.5: Bereiche des Türschilddisplays mit Schriftgrößen, Zeilenlängen und Zeichen-Anzahl.

Bereich	Funktion	Schrifthöhe	Zeilen	Zeichen
Num.	Raumnummer (z.B. "B101")	2 × 133 px	1	6
Info-Zeile 0	Name, Titel	2 × 64 px	1	40
Info-Zeilen 1-3	Stellung, Öffnungszeiten	2 × 36 px	3	40
Nachrichten-Zeilen 0-3	Aktuelle Informationen	1 × 64 px	4	40

Der Zustand des Türschildes wird intern in einer Struktur gespeichert, wobei das Piktogramm durch seinen Index im Bitmap-Array repräsentiert wird. Weiterhin werden die in Tab. 4.6 gelisteten Methoden zur Änderung von Inhalten, zur Übernahme derselben in den EPD-Pixelbuffer und zum vollständigen Neuzeichnen der Bereiche bereitgestellt.

Zu beachten ist, dass die Methoden mit Präfix `roompUpdate...` und `roompDraw...` lediglich in den Pixel-Buffer des EPD-Controllers schreiben, das EPD danach aber nicht updaten, was sinnvoll ist, um das Bild sukzessive im Pixel-Buffer aufzubauen bzw. zu zeichnen. Nachdem dies vollendet ist, muss das Display mit der im nächsten Abschnitt erläuterten Methode `IT8951UpdateDisplay()` geupdated bzw. refresht werden.

Tabelle 4.6: Methoden zur Modifikation der Türschildanzeige.

Methode	Funktion
roompSetRnum(char* str)	Ändern der Raumnummer im Speicher
roompUpdateRnum(void)	Übernahme Raumnummer in den Pixel-Buffer
roompDrawRnum(void)	Neuzeichnen des Raumnummerbereiches im Pixel-Buffer
roompSetInfoLine(int line, char* str)	Ändern einer Info-Zeile im Speicher
roompUpdateInfoLines(void)	Übernahme der Info-Zeilen in den Pixel-Buffer
roompDrawInfo(void)	Neuzeichnen des Info-Bereichs im Pixel-Buffer
roompSetNewsLine(int line, char* str)	Ändern einer Nachrichten-Zeile im Speicher
roompSetNewsPict(int pictIndex)	Ändern des Nachrichten-Piktogramms im Speicher
roompUpdateNewsLines(void)	Übernahme der Nachrichten-Zeilen in den Pixel-Buffer
roompUpdateNewsPict(void)	Übernahme des Piktogramms in den Pixel-Buffer
roompDrawNews(void)	Neuzeichnen des Nachrichten-Bereichs im Pixel-Buffer
roompDrawRoomplate(void)	Neuzeichnen der komplettes Türschild im Pixelbuffer

it8951.h Das `IT8951.h`-Modul bildet die Treiberschnittstelle zum EDP-Controller-Board. Der Code basiert auf dem Beispiel-Code des Display-Herstellers[20] und wurde durch Hr. Ing. Gerhard in [45] ergänzt und optimiert.

Es werden verschiedene Methoden bereitgestellt, z.B. zur Initialisierung des EPD-Controllers, zum Beschreiben des Displays mit Strings, zum Beschreiben mit Bildern, zeichnen von Linien und Rechtecken usw.

Mit zum Treiber gehörend sind die Module `img.h` und `fonts.h`. In Ersterem werden die Piktogramme für den Nachrichtenbereich, sowie das Hochschul-Logo für den Raumnummerbereich deklariert, in Letzterem erfolgt die Deklaration der 3 verwendeten Schriftarten.

Die Definitionen liegen in externen `.c`-Dateien im *epd/font*-Ordner bzw. im *epd/img*-Ordner.

Schriftarten Die drei verfügbaren Schriftarten sind in Tab. 4.7 aufgelistet. Diese sind Bitmap-Arrays, wobei jedes Pixel eines Zeichens mit 4 b kodiert wird, d.h. es werden alle 16 Graustufen des Displays genutzt. Die Schriftarten wurden aus [45] übernommen.

Tabelle 4.7: Verfügbare Schriftarten.

Bezeichnung	Schriftart	Stil	Schrifthöhe	Speicherbedarf
`Large`	Calibri	Normal	133 px, 11,24 mm	442 KiB
`Medium`	Calibri	Fett	64 px, 5,41 mm	102 KiB
`Small`	Calibri	Normal	36 px, 3,04 mm	32 KiB
Gesamt:				577 KiB

In der vorliegenden Anwendung werden aufgrund Anforderung 2b relativ große Schriften benötigt, was aufgrund der Anzahl Graustufen und der kleinen Pixelgröße (s. Absch. 3.1.1) massiv Speicherplatz benötigt.

Um z.B. eine zusätzliche Schriftart mit 2,5 cm Höhe zur Verfügung zu stellen, wären ausgehend von der Größe der Schriftart `Large` 442 KiB$\cdot(25\,\text{mm}/11,24\,\text{mm})^2 =$ 2186 KiB zusätzlicher Flash-Speicher erforderlich und damit mehr als die MCU insgesamt bietet.

Daher wurden die Methoden `IT8951LoadAsciiUpScaled()` und der darauf aufbauenden `IT8951LoadStringUpScaled()` implementiert, die beide als letzten Parameter einen Skalierungsfaktor `uint8_t upscaleFactor` entgegennehmen, nachdem der String bzw. das Zeichen um den entsprechenden Faktor in x- und y-Richtung dupliziert wird. Solange `upscaleFactor` ≤ 3, ergibt sich aufgrund der hohen Displayauflösung nahezu keine sichtbare Unschärfe an den Schriftkanten.

Piktogramme Die Piktogramme sind im Subordner *epd/img* in der Datei `pictograms_320x320_4bpp_0xEbgc.c` definiert. Wie dem Dateinamen zu entnehmen ist, müssen die Piktogramme eine Größe von 320 px × 320 px haben, der Präfix `0xEbgc` indiziert, dass die Hintergrundfarbe der Piktogramme nicht 4 b-weiß (also `0xF`) sondern ein heller Grauton (`0xE`) ist, was der Hintergrundfarbe des Nachrichten-Bereiches des Türschilddisplays entspricht.

Die Piktogramme wurden entsprechend [19] mit dem Bitmap-Converter der Firma Segger[46] erstellt. Um weitere Piktogramme hinzuzufügen, kann dieser Bilder mit 320 px × 320 px einlesen, zu 4 b-Graustufen konvertieren und als `.c`-Datei exportieren. Diese enthält dann ein Pixel-Array, das in die `pictograms_320x320_4bpp_0xEbgc.c` kopiert werden muss, anschließend sind ggf. die Hintergrundfarbe zu ändern[11] und das neue Piktogramm in das `sIMG pictograms[]`-Array am Dateiende einzutragen.

4.2.4 Ressourcenverbrauch der Firmware

Flash-Belegung Wie oben angedeuted werden Flash- und SRAM-Speicherbelegung der MCU vor allem durch die Schrift- und Piktogrammarrays dominiert. Dies soll hier kurz analysiert werden. Abb. 4.2.5 zeigt hierzu die Belegung des Flash-Speichers mit verschiedenen Datentypen.

Die momentan 19 Piktogramme belegen mit 973 KiB knapp die Hälfte des verfügbaren Flash-Speichers, weitere 577 KiB kommen durch die Schriftarten hinzu, sodass

[11] Dies kann z.B. über die *Search and Replace* Funktion eines beliebigen Texteditors erfolgen, indem jedes `F` durch ein `E` ersetzt wird.

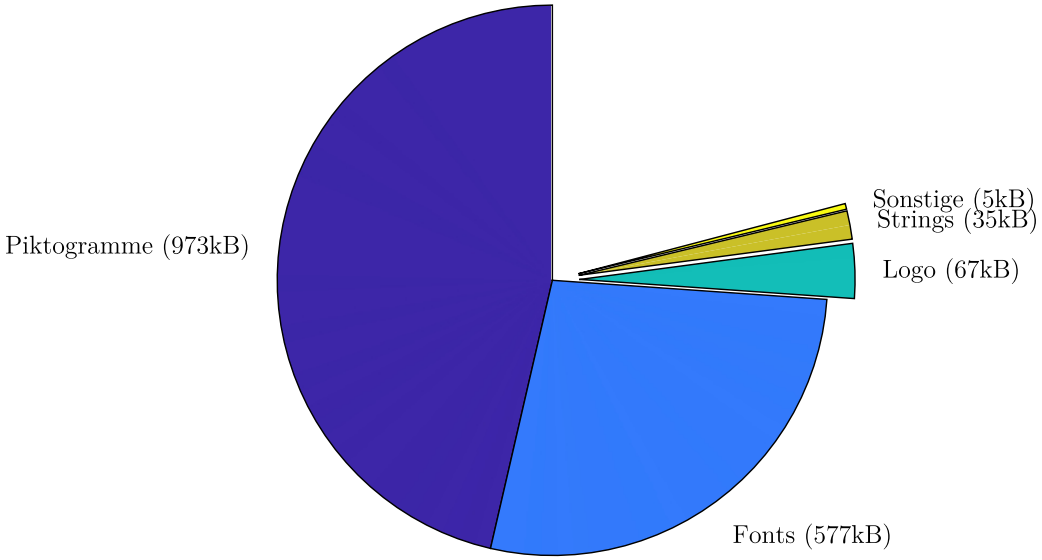


Abbildung 4.2.5: Flash-Speicherbelegung des Mikrocontrollers.

der Speicher hiermit bereits zu knapp 75% belegt ist. Die übrigen Daten, vor allem Strings und das Hochschul-Logo belegen zusammen weitere 107 KiB, sodass letztlich noch 22% des Speichers frei bleiben.

SRAM-Belegung Abb. 4.2.6 zeigt die Belegung des SRAM-Speichers durch statisch allokierten Speicher sowie den Stack. Auf die Verwendung dynamisch allokierten Speichers wurde verzichtet.

Der Frame-Buffer des `IT8951.h`-Moduls belegt mit 142 KiB mehr als die Hälfte des SRAMs, durch die übrigen Variablen kommen weitere 6 KiB hinzu, sodass sich eine Belegung von insgesamt 56% ergibt. Der Frame-Buffer (`uint16_t gpFrame16Buf[]`) dient im Display-Treiber-Modul dazu, den an den Display-Controller zu übermittelnden Bildbereich (z.B. eine Textzeile oder ein Piktogramm) zunächst lokal aufzubauen und dann an den Display-Controller zu senden, was notwendig ist, da bisher keine Möglichkeit gefunden werden konnte, Daten aus dem Pixel-Buffer des Controllers auszulesen, was aber teilweise für die entsprechenden Display-Methoden zum schreiben von Text und Bildern notwendig ist.

Damit ergibt sich dass der Pixel-Buffer im Controller nur ausschnittsweise, mit maximal $142\,\mathrm{KiB} \cdot 2\,\mathrm{px} \cdot \mathrm{B}^{-1} = 284 \cdot 10^3\,\mathrm{px}$ beschrieben werden kann, was praktisch

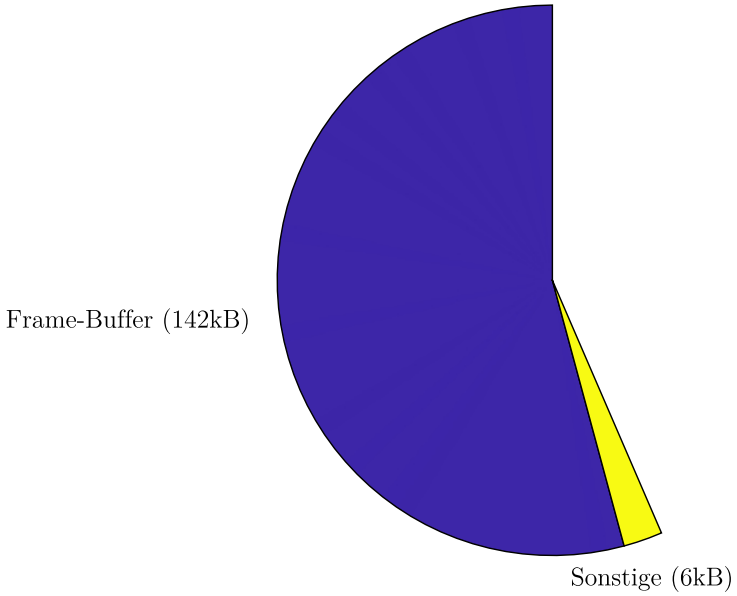


Abbildung 4.2.6: SRAM-Speicherbelegung des Mikrocontrollers.

aber keine Nachteile ergibt, da der Pixel-Buffer des Controllers beliebig oft vor einem Refresh modifiziert werden kann.

4.2.5 Main-Programmablauf

Abschließend soll nun der Programmablauf in der Main-Schleife der Firmware mit dem zugehörigen NFC-Interrupt beleuchtet werden. Hierzu sind in Abb. 4.2.7 die Programmablaufpläne der `main()`-Funktion und der `PORT3_IRQHandler()` dargestellt. Letzterer stellt die Interrupt-Service-Routine für den GPIO-Port 3 dar, d.h. jener Port, an dem an Pin `P3.5` der GPO-Interrupt des NFC-Transceivers eintrifft (s. Absch. 4.1.2).

Die Programmflussdiagramme abstrahieren den Code dabei stark, was der Übersichtlichkeit und Verständlichkeit dienen soll.[12]

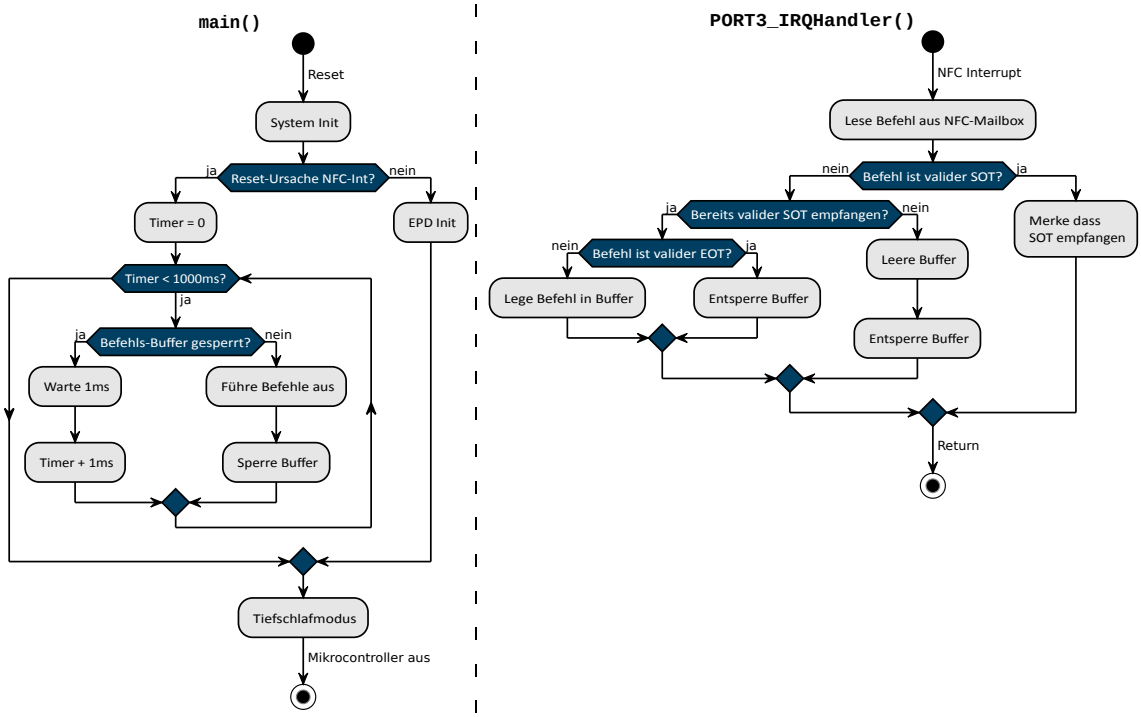


Abbildung 4.2.7: Main Programmablauf und zugehörige NFC-ISR.

Beim erstmaligen Start, d.h. *Power-On-Reset* der MCU, initialisiert sich diese zunächst, d.h. die Taktrate wird auf 48 MHz hochgesetzt und die Pins werden initialisiert. Da die Resetursache kein NFC-Interrupt war, wird das Display zunächst als Muster-Türschild initialisiert, was hauptsächlich für die Evaluation nützlich war. Danach begibt sich die MCU in den Tiefschlafmodus `LPM 4.5`.

Hier behalten lediglich die Pinregister und damit Pins ihren Zustand, außerdem ist ein Teil der GPIO-Interrupt-Logik aktiv, ansonsten sind alle Oszillatoren, Speicher und die gesamte Peripherie ausgeschaltet bzw. spannungsfrei. Ein GPIO-

[12] Für eine codenähere Darstellung des Interrupt-Handlers sei auf Abb. 4.2.3 verwiesen.

Interrupt im `LPM 4.5` führt damit zu einem Reset des Controllers, d.h. dieser kann aus dem `LPM 4.5` nicht direkt aufwachen und im Programmfluss fortfahren.

Trifft nun ein Interrupt-Signal vom NFC-Transceiver ein, wird die MCU also resettet und erneut initialisiert. Da die Reset-Ursache nun ein NFC-Interrupt an Port 3 war, wird eine Schleife betreten, die überprüft, ob der Befehls-Buffer noch immer lesegesperrt ist. Danach wird für 1 ms gewartet, anschließend erfolgt die nächste Iteration. Dies wird bis zu 1000 mal wiederholt, womit sich eine Gesamtwartezeit von 1 s ergibt.

In dieser Zeit muss nun die NFC-Kommunikation abgewickelt werden. Während das Hauptprogramm durch die o.g. Schleife iteriert, treffen immer wieder NFC-Interrupts ein, wenn ein neuer Befehl in der NFC-Mailbox liegt. Da nichtmehr im Tiefschlaf, betritt die MCU nun den `PORT3_IRQHandler`.

Es wird zunächst der Befehl aus der Mailbox gelesen, damit diese für den nächsten Befehl frei ist. Anschließend wird überprüft ob der Befehl ein gültiger `SOT`-Rahmen (Start-Of-Transmission, s. Absch. 4.2.2) ist, falls nein wird überprüft ob denn bereits ein gültiger `SOT`-Rahmen empfangen wurde, falls nein wird der Befehlsbuffer vor dem return geleert und entsperrt.

Wurde hingegen bereits ein gültiger `SOT`-Rahmen empfangen, wird überprüft ob nun ein gültiger `EOT`-Rahmen (End-Of-Transmission) empfangen wurde, falls nein, wird der empfangene Befehl in den Befehls-Buffer gelegt, falls ja ist die Übertragung vollständig und der Befehlsbuffer wird zur Abarbeitung durch die `main()`-Funktion entsperrt.

Ist in der Warteschleife also nun der Befehl-Buffer entsperrt, werden die Befehle im Buffer abgearbeitet, anschließend wird der Buffer wieder gesperrt, und nachdem der Timer 1000 ms erreicht hat, begibt sich die MCU wieder in den Tiefschlafmodus.

4.3 Implementierung der App

Im Folgenden soll die Implementierung der Android-App erläutert werden. Da ein Großteil derselben generisch ist, soll dies nur überblicksweise geschehen und tiefergehende Erläuterungen auf das für die vorliegende Anwendung wichtigste Modul, den `EInkCommandTransmitter` beschränkt werden.

Die Programmierung der App erfolgte in Java, die Entwicklung unter Verwendung der Android Studio IDE.

Es wird nun zunächst die grafische Benutzeroberfläche der App vorgestellt, dann soll die Software-Architektur mit ihren Modulen erläutert werden. Abschließend wird

die o.g. Transmitter-Klasse beleuchtet, welche die NFC-Kommunikation abwickelt und das Skeid-Protokoll App-seitig implementiert.

4.3.1 Benutzeroberfläche der App

Abb. 4.3.1 zeigt im linken Teilbild das Hauptfenster der Benutzeroberfläche der App. Mit einem Fingerdruck (im Folgenden *Klick*) auf den +-Button, wechselt die Ansicht zum in der mittleren Anschicht dargestellten Fenster, in dem sich neue Abwesendheitsnachrichten anlegen, mit einem Piktogramm versehen und speichern lassen. Diese stehen dann, wie im rechten Teilbild dargestellt, im Hauptfenster zur Verfügung. Mit kurzem Klick auf eine Nachricht lässt sich diese dann an das Türschild senden, bei langem Klick erscheinen zwei Buttons in der oberen grünen Leiste, mit denen sich die ausgewählte Nachricht löschen oder bearbeiten lässt.

Alternativ lassen sich im Eingabebereich ganz unten im Fenster, wie im rechten Teilbild ebenfalls zu sehen, direkt Nachrichten eingeben und mit Klick auf den Send-Button verschicken. Die eingegebene Nachricht wird nicht gespeichert.

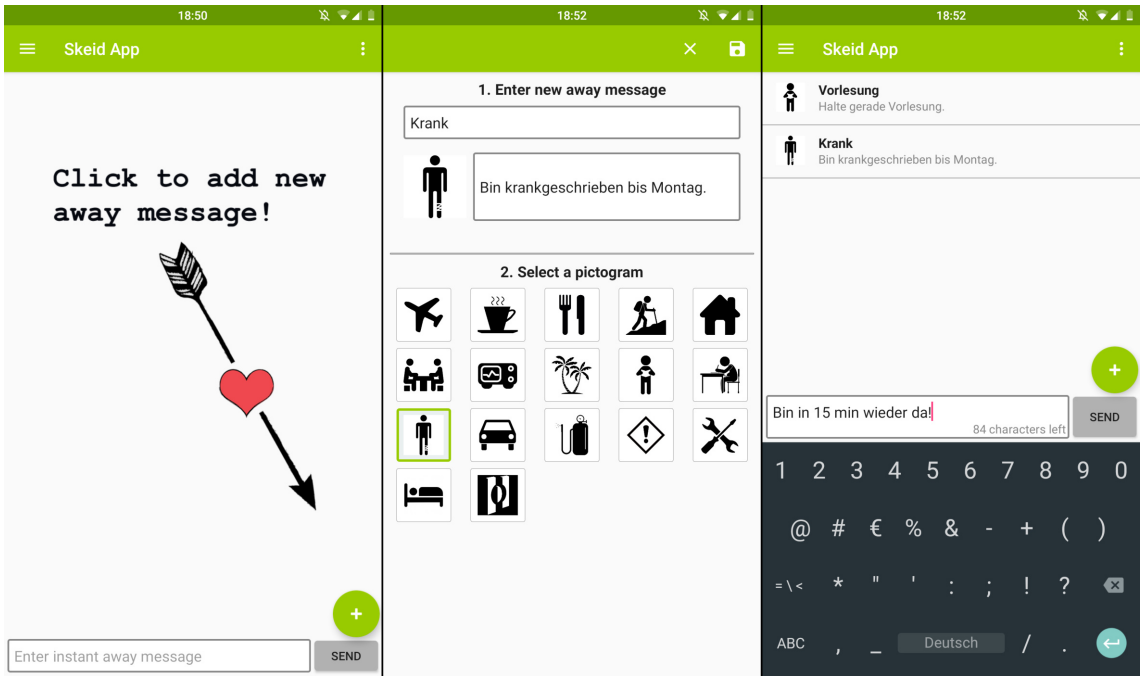


Abbildung 4.3.1: Hauptfenster der App leer (links) und mit gespeicherten Nachrichten (rechts), sowie Eingabemenü für neue Nachrichten (mittig).

Das Benutzerfeedback zum Senden der Nachricht erfolgt mittels der drei in Abb. 4.3.2 dargestellten Dialoge. Nach Klick auf eine gespeicherte Nachricht bzw. nach Klick auf den Send-Button, öffnet sich der im linken Teilbild dargestellte Dialog, der den Benutzer auffordert, das Smartphon zur NFC-Kommunikation an das

Türschild zu halten. Sobald der Kontakt mit dem Türschild hergestellt ist, vibriert das Smartphone, als zusätzliches Feedbacksignal.

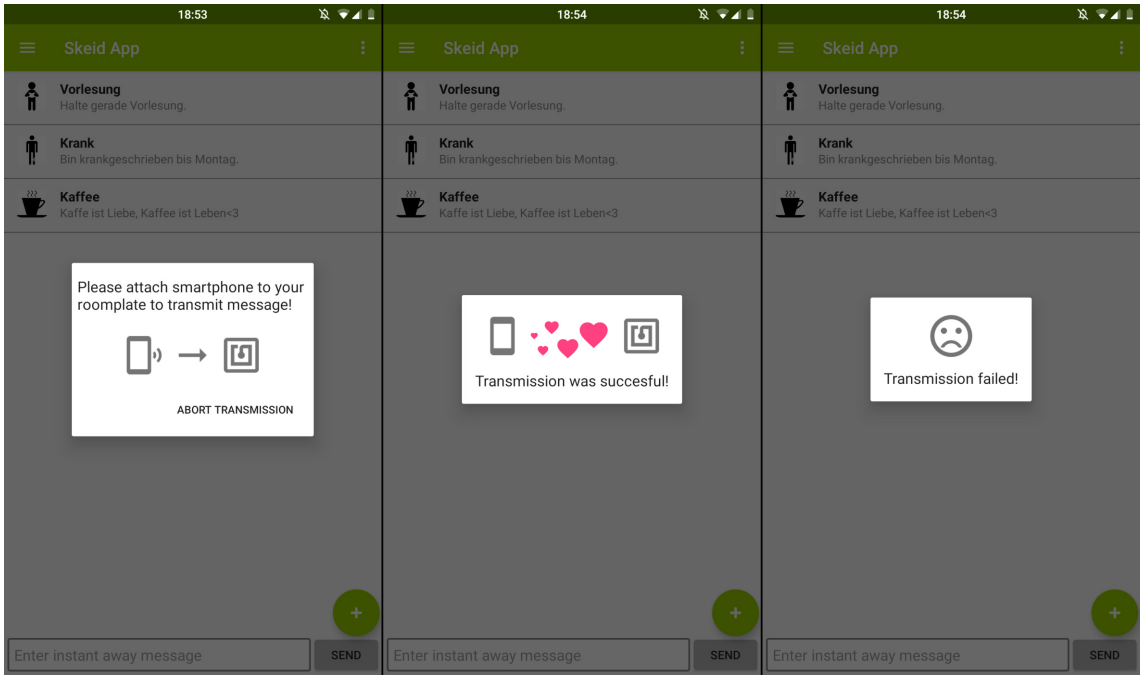


Abbildung 4.3.2: Die drei Sendedialoge der App. Connecting- (links), Success- (mittig) und Failure-Dialog (rechts).

Nach dem erfolgreichen Senden der Nachricht wird der im mittleren Teilbild dargestellte Dialog angezeigt, um dem Benutzer mitzuteilen, dass das Smartphone nun vom Türschild genommen werden kann. Schlägt die Übertragung fehl, z.B. durch einen Timeout wenn das Türschild nichtmehr erreichbar ist, wird der im rechten Teilbild dargestellte Dialog angezeigt.

4.3.2 Softwarearchitektur der App

Begriffe Zunächst sind einige Begriffe bzw. Java-Klassen aus dem Android-Umfeld zu definieren:

- **Activity** Eine Activity ist eine einzelne, fokussierte Sache (i.d.R. ein Fenster) [47]. In der vorliegenden App sind zwei Activities vorhanden, eine die das Hauptfenster enthält bzw. implementiert (linkes Teilbild Abb. 4.3.1) und eine die das Menü zum Anlegen neuer Nachrichten implementiert. Mit dem Starten, Pausieren, Wiederaufrufen oder Beenden einer Activity werden vom Android-System bestimmte Methoden aufgerufen, welche die Activity implementieren muss.

- **Fragment** Ein Fragment repräsentiert ein bestimmtes Verhalten oder einen Teil der Benutzer-Schnittstelle einer Activity[48]. Die in Abb. 4.3.2 dargestellten Dialoge sind z.B. als Fragments implementiert. Wie auch die Activities müssen diese bestimmte Methoden implementieren, die vom Android-System aufgerufen werden.

Softwarearchitektur In Abb. 4.3.3 ist die Software-Architektur der App dargestellt. Die einzelnen Klassen sind in blau, die Packages in grau hinterlegt. Nach Packages sortiert werden nun einzelne Komponenten erläutert.

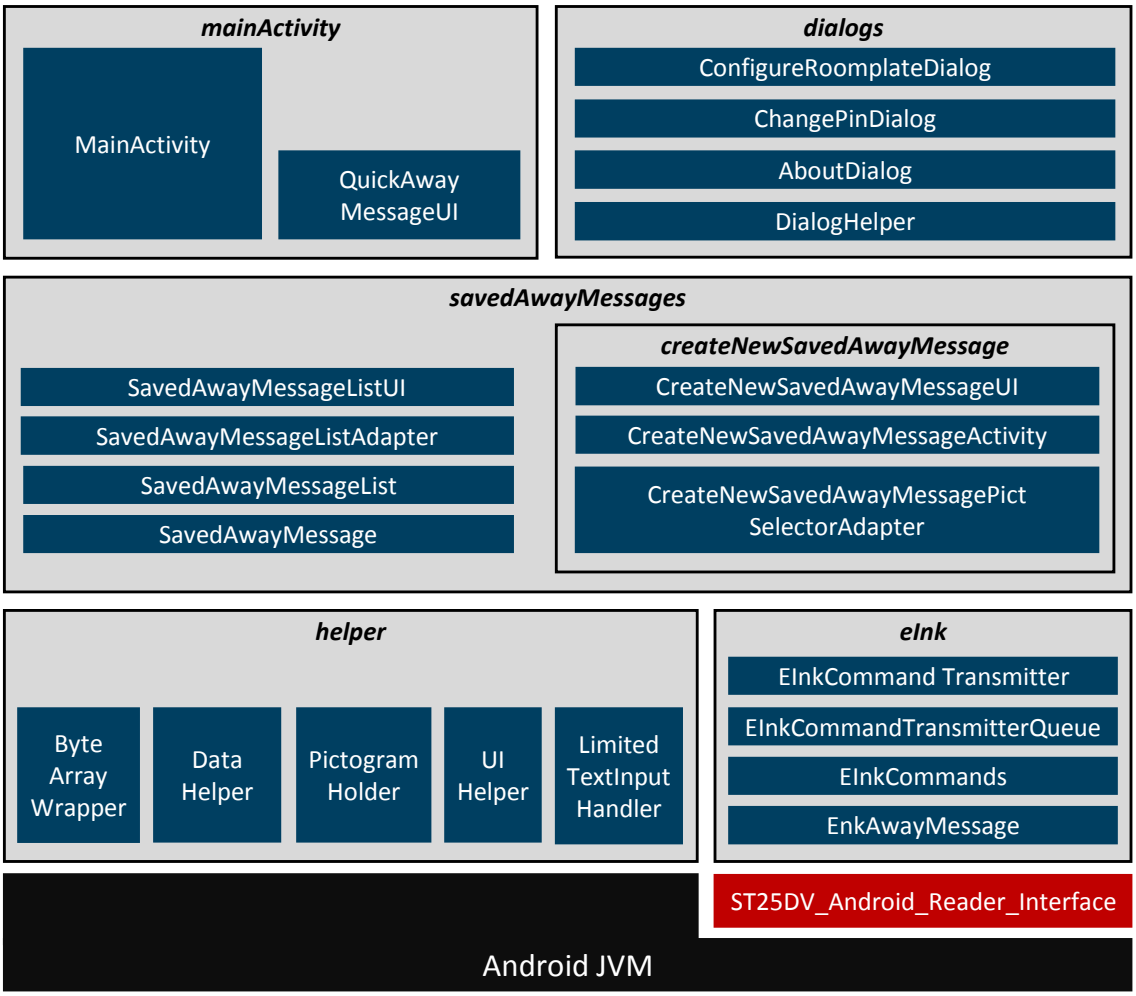


Abbildung 4.3.3: Softwarearchitektur der Android-App mit Klassen in blau und Packages in grau.

- ***mainActivity*** Enthält die `MainActivity`, die wie angesprochen das Hauptfenster enthält, sowie das `QuickAwayMessageUI`, welches die Klasse Fragment erweitert, und die Eingabezeile zum direkten Senden von Nachrichten implementiert.

- ***savedAwayMessages*** Enthält Klassen die in der `mainActivity` anzuzeigende Liste mit den zugehörigen Funktionalitäten implementieren. Das Subpackage *createNewSavesAwayMessage* dient zum Anlegen bzw. Bearbeiten neuer Abwesendheitsnachrichten und wird separat erläutert.

  Die `CreateNewSavedAwayMessageUI` erweitert die Fragment-Klasse und dient zur Darstellung der Nachrichtenliste im Hauptfenster der `mainActivity`. Es enthält dazu ein `SavedAwayMessageList`-Objekt, welches die eigentliche Liste implementiert und verschiedene Methoden z.B. zum Hinufügen neuer Nachrichten oder zum Speichern der Liste in den *SharedPreferences* des Android-Systems bereitstellt.

  Der `SavedAwayMessageAdapter` erweitert die `RecyclerView.Adapter`-Klasse und dient einerseits zur Darstellung eines Elementes der Liste und weiterhin als Schnittstelle zwischen der `CreateNewSavedAwayMessageUI`-Klasse und der einzelnen Nachricht, d.h. hier ist definiert, was beim kurzem oder langem Klick auf eine bestimmte Nachricht der Liste geschehen soll. Hierzu muss sie bestimmte Methoden implementieren, die dann z.B. beim Klick auf eine Nachricht aufgerufen werden. Hier geschieht somit auch das Senden der Nachricht.

- ***createNewSavedAwayMessage*** Enthält mit der `CreateNewSavedAwayMessageActivity` die Activity zum Erstellen neuer Nachrichten. Das eigentliche Layout des Fensters geschieht durch die `CreateNewSavedAwayMessageUI`-Klasse. Hier ist auch die Piktogramm-Liste zur Auswahl eines Piktogramms implementiert. Als Schnittstelle zwischen ausgewähltem Bild (d.h. was beim Klick auf ein Piktogramm geschehen soll) und der `CreateNewSavedAwayMessageUI`-Klasse dient ähnlich wie oben der `CreateNewSavedAwayMessagePictSelectorAdapter`.

- ***eInk*** Enthält Klassen zur Abstraktion eines Transmitters, der das Skeid-Protokoll implementiert. Die `EInkCommandTransmitter`-Klasse implementiert dabei letztlich den Transmitter. Diesem wird das nachfolgende Kapitel gewidmet.

  `EInkAwayMessage` abstrahiert dabei die Nachrichten, die `EInkCommands`-Klasse implementiert Methoden zum Senden der im Skeid-Protokoll definierten Befehle. Zum Senden eines SOT-Befehls muss z.B. `EInkCommands.presentStartCmd(String pin);` aufgerufen werden, wobei `pin` die Pin als vierstelliger, nullterminierter ASCII-String ist.

  Die `EInkCommandTransmitterQueue` implementiert eine Queue, in denen die Befehle zunächst abgelegt werden, um sukzessive durch den `EInkCommand-`

`Transmitter` gesendet zu werden. Hierzu wird auf das `ST25DV_Android_Reader_Interface` zurückgegriffen, welches Methoden und Klassen zum Verbindungsaufbau und Kommunikation mit dem ST25DV über die NFC-Schnittstelle des Smartphones bereitstellt und somit die Reader-Funktion durch das Smartphone bereistellt. Es handelt sich dabei nicht um ein Java-Interface, sondern eine Android-Bibliothek, die vom Hersteller des `ST25DV`-Transceiver-ICs bereitgestellt wird und Teil der `ST25SDK` ist[43].

4.3.3 Tranmitter-Klasse

Hier soll die Implementierung der `EInkCommandTransmitter`-Klasse untersucht werden, die, wie oben angesprochen, dafür zuständig ist, die Nachrichten ans Türschild zu übertragen, bzw. die NFC-Kommunikation abzuwickeln.

Der Transmitter implementiert das im `ST25SDK` definierte `TagDiscovery.onTagDiscoveryCompletedListener`-Interface und ist damit in der Lage eine NFC-Verbindung zum `ST25DV` aufzubauen[43, S. 12ff.]. Zur Implementierung des Interfaces muss der Transmitter damit auch die Android-Klasse `AsyncTask` erweitern, welche einen nebenläufigen Task erzeugt. Konkret heißt das, dass der Transmitter die Methode `doInBackground()` implementieren muss, die nebenläufig ausgeführt wird, d.h. in dieser Methode geschieht letztlich der Verbindungsaufbau und die NFC-Kommunikation.

Es darf zeitgleich nur ein Transmitterobjekt existieren, daher wird dieses mit dem Starten der App erzeugt. Die `doInBackground`-Methode ist dabei eine Endlosschleife die einen simplen endlichen Zustandsautomaten implementiert. Abhängig vom jeweiligen Zustand werden die entsprechenden in Abb. 4.3.2 dargestellten Dialoge angezeigt. Die Zustände `IDLE`, `ABORTING` und `TRANSMITTING` sind Dialog-frei.

In Abb. 4.3.4 ist das Programmflussdiagramm der Implementierung dargestellt, die entsprecheden Transmitter-Zustände werden im Folgenden erläutert.

- **IDLE** Es wird überprüft, ob Nachrichten zum Senden in der Queue bereitliegen und gegebenenfalls in den `CONNECTING`-Zustand zum Verbindungsaufbau gewechselt. Liegen keine Nachrichten bereit, wird der Thread für 50 ms pausiert, danach erfolgt die nächste Iteration.

- **CONNECTING** Zunächst wird überprüft, ob der Abort-Button im Sendedialog zum Abbruch der Übertragung gedrückt wurde, und gegebenenfalls in den `ABORT`-Zustand gewechselt. Ansonsten wird versucht eine Verbindung zum `ST25DV` aufzubauen, falls dies erfolgreich ist, wird in den `TRANSMITTING`-

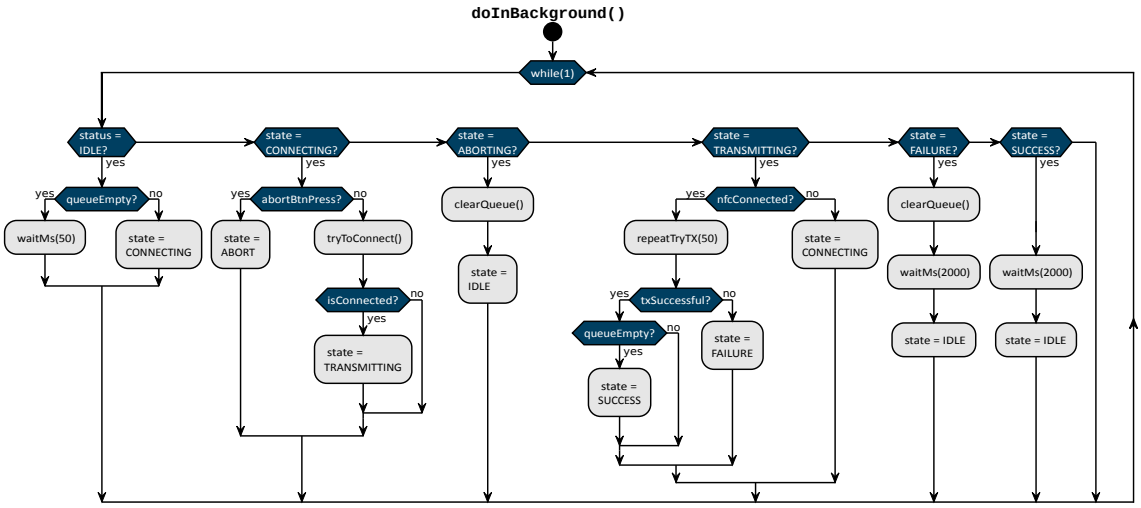


Abbildung 4.3.4: Programmflussdiagramm der Transmitter-Schleife.

Zustand gewechselt, ansonsten wird in der nächsten Iteration erneut versucht die Verbindung herzustellen.

- **ABORTING** Ist ein Zwischenzustand, in dem lediglich die Queue geleert wird. Anschließend geht der Transmitter in den `IDLE`-Zustand.

- **TRANSMITTING** Hier wird vor jeder Iteration zunächst die Verbindung überprüft und gegebenenfalls in den `CONNECTING`-Zustand zum Verbindungsaufbau gewechselt. Anschließend wird, angedeuted durch `repeatTryTX(50)`, in einer Schleife mittels eines `try/catch`-Konstruktes bis zu 50 mal versucht eine Nachricht zu senden. Ist die Mailbox des `ST25DV` noch belegt oder wird gerade MCU-seitig über die I^2C-Schnittstelle auf diesen zugegriffen, wirft die `ST25DVTag.writeMailboxMessage(msg)`-Methode eine `Exception`, die einen Zähler inkrementiert. Nach Verlassen der Schleife wird überprüft, ob der Zähler einen Wert kleiner 50 hat, falls nein ist die Übertragung 50 mal gescheitert, sodass der Transmitter in den `FAILURE`-Zustand geht. War die Übertragung erfolgreich, wird abhängig davon, ob weitere Nachrichten zu senden sind im `TRANSMITTING`-Zustand verblieben, oder in den `SUCCESS`-Zustand gewechselt.

- **FAILURE** zunächst wird die Queue geleert, anschließend wird 2 s gewartet um den Failure-Dialog anzuzeigen. Danach wird in den `IDLE`-Zustand gewechselt.

- **SUCCESS** Da die Queue bereits leer ist, wird lediglich 2 s zur Anzeige des Success-Dialoges gewartet, danach wird ebenfalls in den `IDLE`-Zustand gewechselt.

Kapitel 5

Ergebnisse

In diesem Kapitel sollen die am Eval-Board durchgeführten Messungen erläutert werden, um im nachfolgenden Kapitel zu klären inwieweit die Systemimplementierung den Anforderungen genügt und um Verbesserungspotenziale aufzufinden.

5.1 Energieversorgung

Wie in Absch. 4.1.3 angesprochen, sind auf dem Eval-Board zwei Spannungswandler zur Versorgung des Display-Controllers vorhanden. Es sollte überprüft werden, ob sich dieser, unter Überbrückung des auf dem Controller-Boards vorhandenen $3,3\,\mathrm{V}$-Wandlers, effizienter direkt aus $3,3\,\mathrm{V}$ versorgen lässt, als durch eine $5\,\mathrm{V}$-Versorgung.

Es zeigte sich hierbei, dass das Controller-Board des 7,8”-Displays von WaveShare, im Gegensatz zum Controller-Board des 9,7”-Displays nicht in der Lage ist, das Display aus $3,3\,\mathrm{V}$ zu refreshen. Zwar lässt sich der Controller initilisieren und der Pixel-Buffer beschreiben, der Refresh aber hängt sich der Controller auf. Da sich dieses Verhalten auch bei der $3,3\,\mathrm{V}$-Versorgung durch ein Labornetzteil zeigte, musste der Ansatz verworfen werden, sodass das Controller-Board im Folgenden stets mit $5\,\mathrm{V}$ versorgt wurde.

5.2 Verfügbare Energie

Zunächst wurde die durch die Solarzelle und Energy-Harvester verfügbare Energie in Abhängigkeit von der Belechtungsstärke bestimmt.

Hierzu wurde das Funktionsmuster nach dem in Abb. 5.2.1 gezeigten Schema in verschiedene Beleuchtungssituationen gebracht und die Beleuchtungsstärke mit einem Lux-Meter gemessen, sowie die Ausgangsleistung der Solarzellen P_Z sowie die Ladeleistung des Akkus P_A mittels Multimeter gemessen. Abb. 5.2.2 zeigt den

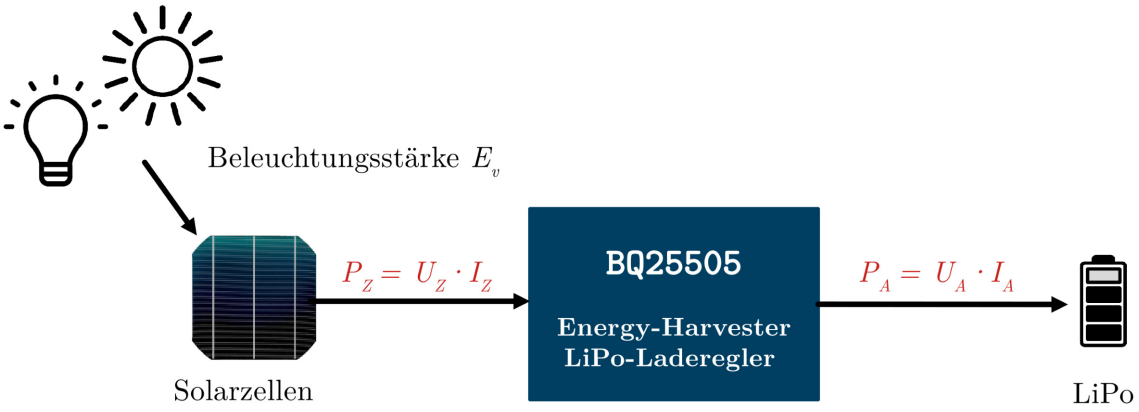


Abbildung 5.2.1: Schematischer Messaufbau zur Bestimmung der verfügbaren Energie.

Abbildung 5.2.2: Realisierter Messaufbau zur Bestimmung der verfügbaren Energie.

Messaufbau, rechts über dem Türschild ist die Luxmeter-Sonde zu sehen. Die Ergebnisse sind in Abb. 5.2.3 dargestellt.

Die Leistungen P_Z und P_A steigen bei geringen Beleuchtungsstärken E_v von bis zu einigen hundert Lux linear mit E_v an. P_Z steigt mit etwa $9,3\,\mu\mathrm{W\ lx^{-1}}$ an, entsprechend ergibt sich bei einer Gesamtzellfläche von $218,5\,\mathrm{cm^2}$ ein flächenbezogener Wert von $42,7\,\mathrm{nW\ cm^{-2}\ lx^{-1}}$ was in der Größenordnung des in Absch. 3.3.1 errechneten Wertes von $60\,\mathrm{nW\ cm^{-2}\ lx^{-1}}$ liegt.

In schwarz abgetragen ist der Ladewirkungsgrad, berechnet als $\eta = P_\mathrm{A}/P_\mathrm{Z}$. Dieser erreicht bereits bei $E_v = 10\,\mathrm{lx}$ knapp 80% und steigt anschließend bis auf 90%.

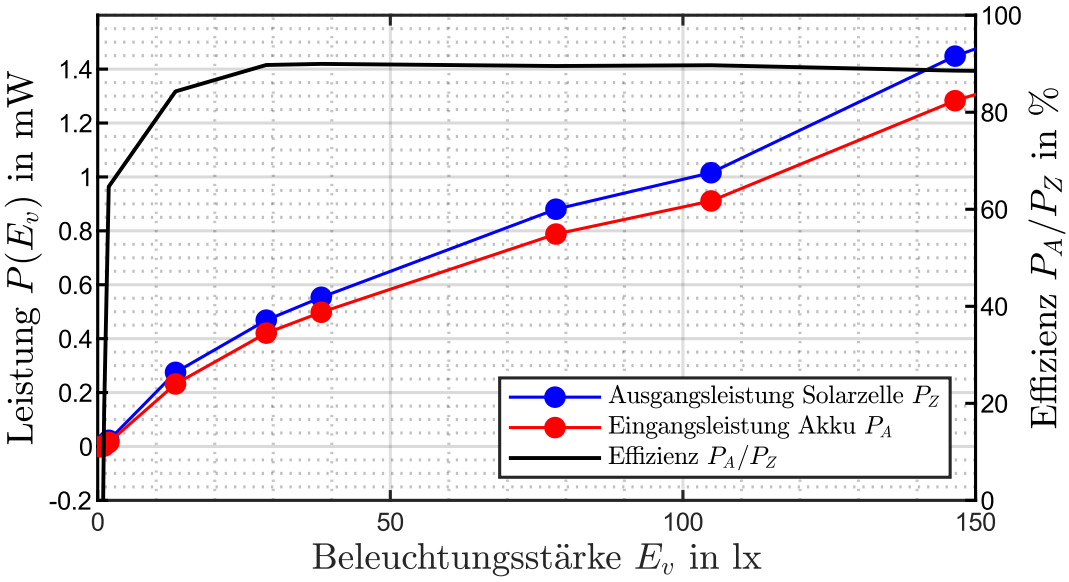


Abbildung 5.2.3: Messergebnisse zur Bestimmung der verfügbaren Energie.

Mit zunehmender Beleuchtungsstärke bzw. Ladeleistung sinkt dieser wieder etwas, was daran liegt, dass der `BQ25505`-Harvester-IC für Leistungen im Mikrowattbereich optimiert ist.

5.3 Verschaltung der Solarzellen

Die im letzten Abschnitt erläuterte Messung wurde einmal mit 20 Zellen durchgeführt, sowie ein weiteres Mal, bei dem nur 5 der 20 Zellen durch die Jumper mit dem Harvester-IC verbunden waren. Es sollte nun untersucht werden, ob die geerntete Leistung bei identischen Beleuchtungsstärken E_v tatsächlich um den zu erwarteten Faktor $20/5 = 4$ geringer war. Damit soll geklärt werden, ob das Parallelverschalten derart vieler Zellen ohne Blockingdioden der Annahme entsprechend unbedenklich ist. Abb. 5.3.1 zeigt die zugehörigen Messergebnisse in Form der Quotienten $P20_\mathrm{Z}/P5_\mathrm{Z}$ und $P20_\mathrm{A}/P5_\mathrm{A}$ in Abhängigkeit von E_v.

Bei Beleuchtungsstärken von $E_v > 80\,\mathrm{lx}$ zeigt sich, dass beide Quotienten kleiner als 4 sind, damit ist die im 20-Zellbetrieb geerntete Leistung kleiner als das vierfache der im 5-Zellbetrieb geernteten Leistung. Hieraus folgt, dass bei $E_v > 80\,\mathrm{lx}$ jede zusätzlich hinzukommende Zelle die Gesamtleistung zwar weiter erhöht, jedoch in immer geringerem Maße, ihr *Grenzertrag* ist sinkend.

Bei Beleuchtungsstärken kleiner als $E_v = 80\,\mathrm{lx}$ zeigt sich überraschenderweise, dass dass die Quotienten wesentlich größer als 4 werden, womit z.B. bei $E_v = 10\,\mathrm{lx}$ mit 20 Zellen das 10-fache der Leistung als im 5-Zellbetrieb geerntet werden.

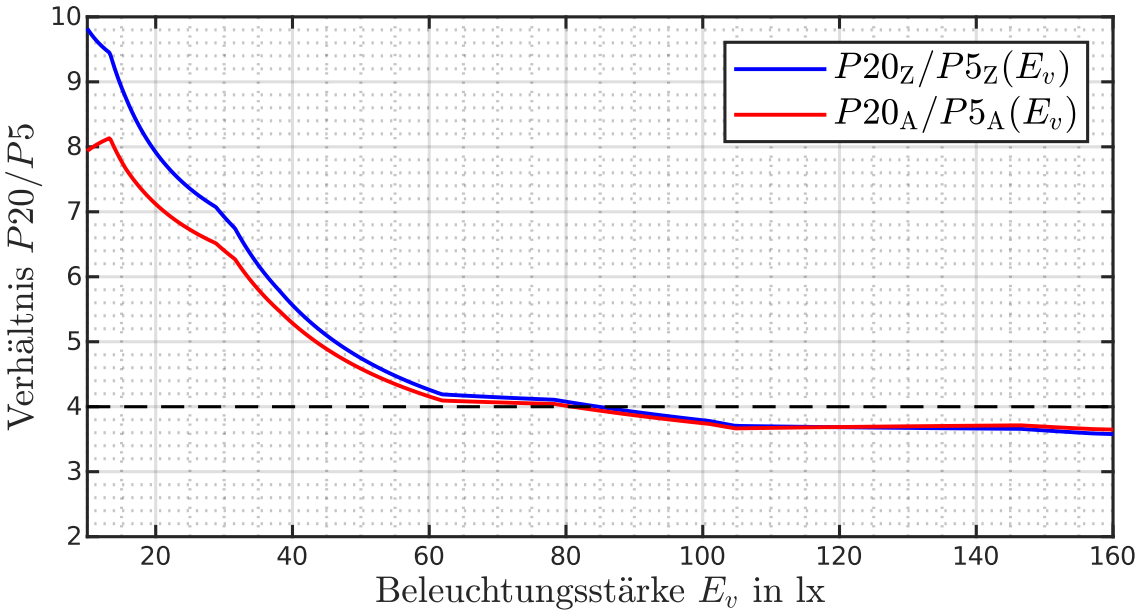


Abbildung 5.3.1: Messergebnisse zum Vergleich bei 20- und 5-Zell-Betrieb.

Zu erklären ist dies damit, dass mögliche Verluste durch Querströme in diesem Bereich im Vergleich zu der mit der Eingangsleistung sinkenden Wandlungseffizienz und Problemen mit dem MPPT vernachlässigbar klein sind. Durch die höhere Leistung im 20-Zellbetrieb, kann der `BQ25505` die Leistung wesentlich effizienter ernten, wodurch überproportional mehr Leistung entnommen werden kann. Ab $E_v > 80\,\mathrm{lx}$ relativiert sich dieser Effekt offenbar.

Zum Vergleich der beiden Kurven ist zu sagen, dass bei schwacher Beleuchtung der Quotient $P20_\mathrm{A}/P5_\mathrm{A}$ der Ladeleistungen kleiner ist, als der Quotient der Ausgangsleistungen der Solarzellen, was durch zusätzliche Ladeverluste zu begründen ist, die sich mit steigender Beleuchtungsstärke relativieren.

5.4 Energieverbrauch

Im Folgenden wird der Energieverbrauch des Systems bzw. der Systemkomponenten untersucht. Die Messungen wurden mit einer Präzisions-*Source-Measure-Unit* (SMU) durchgeführt.

5.4.1 Gesamtverbrauch während Update

Zunächst wurde der Gesamtverbrauch des Systems während eines Updates bestimmt. Hierzu wurde der LiPo abgeklemmt und anstelle desselben die SMU kontaktiert. Diese wurde auf die Nennspannung des LiPos von $3,7\,\mathrm{V}$ eingestellt, mit

einem Stromlimit von 3 A. Anschließend wurde ausgehend vom Schlafzustand des Systems eine Nachricht übertragen und ein Displayupdate durchgeführt. Die Messwerte wurden von der SMU mit einer Periodendauer von 150 µs, d.h. mit 6, 67 kHz aufgezeichnet. Das Ergebnis ist in Abb. 5.4.1 dargestellt.

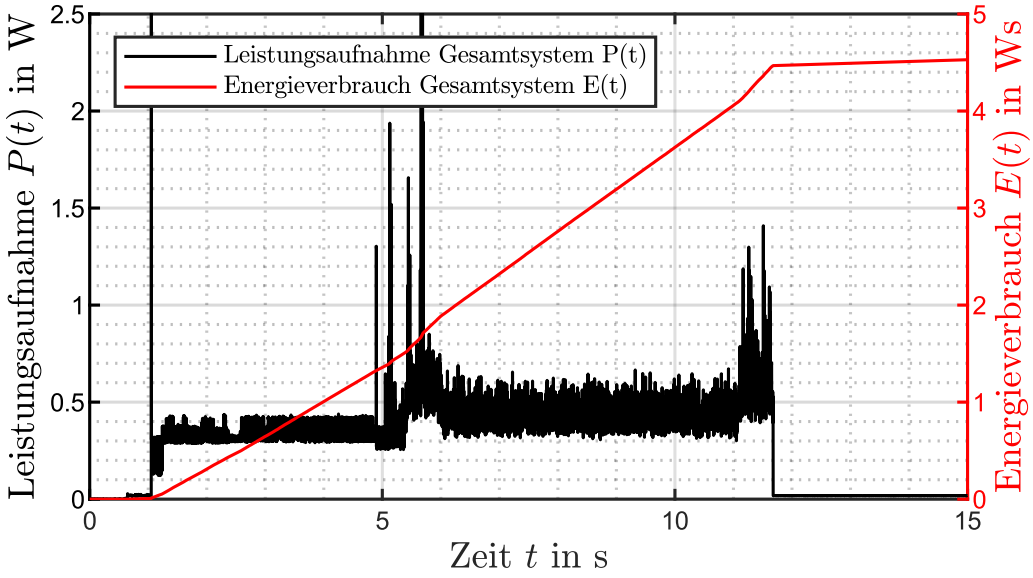


Abbildung 5.4.1: Messergebnisse zum Gesamtverbrauch des Systems.

Bis $t = 5\,\mathrm{s}$ wird das Display mit weiß überschrieben, anschließend folgt der erste Refresh, was durch die Leistungsspitzen ab $t = 5\,\mathrm{s}$ gekennzeichnet ist. Dann erfolgt die Übertragung des eigentlichen Bildes, bei $t = 11\,\mathrm{s}$ der zweite Refresh. Alles in Allem ist die Leistungsaufnahme nahezu identisch mit der in Absch. 3.1.2 durchgeführten Messung, d.h. im Wachzustand ist der Verbrauch der übrigen Systemkomponenten im Vergleich zum Verbrauch des Displays vollkommen zu vernachlässigen. Alleine für die SPI-Kommunikation benötigt der Controller nahezu durchgehend über 300 mW, die sich über die Updatezeit auf einen Energieverbrauch von mehr als 4, 5 Ws kumulieren. Damit ist der Verbrauch während eines Updates etwa um das 750-fache größer als die Leistung der Solarzellen bei einer guten Beleuchtung von 20 lx.

5.4.2 Energieverbrauch im Schlafmodus

Um die Leistungsaufnahme des Systems im Schlafmodus zu bestimmen wurde ein zum vorherigen Abschintt identischer Messaufbau gewählt. Das System wurde anstelle des LiPos bei 3, 7 V aus der SMU versorgt und die Leistungsaufnahme mit einer Periode von 20 µs aufgezeichnet. Die Ergebnisse sind in Abb. 5.4.2 zu sehen.

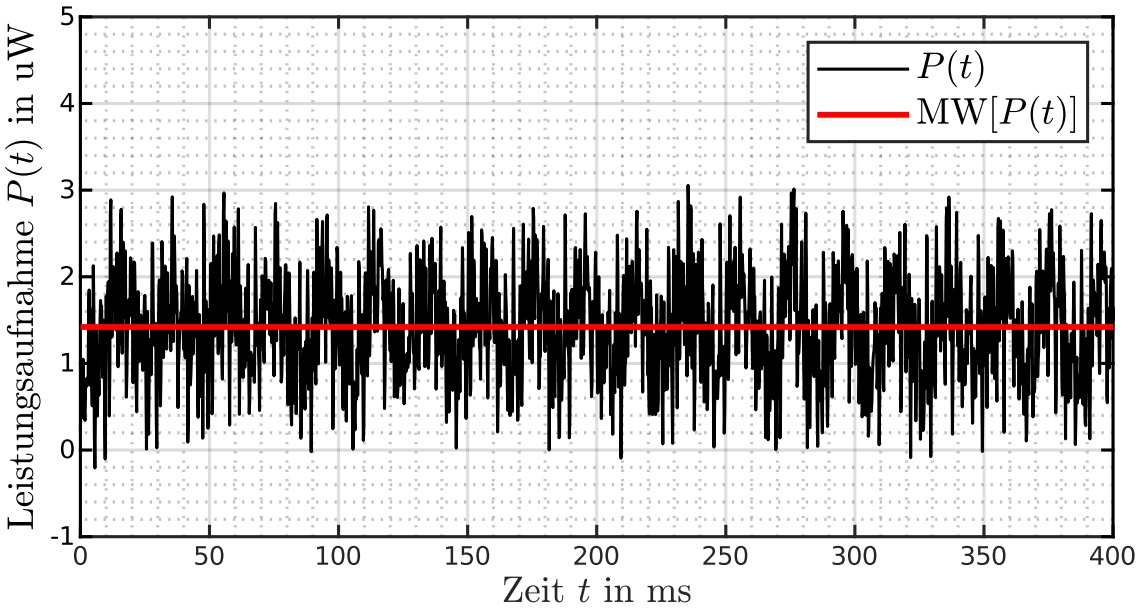


Abbildung 5.4.2: Messergebnisse zum Energieverbrauch des Systems im Schlafzustand.

Bei genauer Betrachtung ist zu erkennen, dass das in schwarz gezeichnete Messergebnis der Leistungsaufnahme $P(t)$ mit einem starken 50 Hz-Rauschen behaftet ist. Darum wurde der in rot dargestellte Mittelwert MW$[P(t)]$ der Leistungsaufnahme berechnet, der bei etwa $1,5\,\mu W$ liegt. Somit ergibt sich bei $3,7\,V$ eine mittlere Stromaufnahme von lediglich 405 nA.

5.4.3 Verbrauch Systemkomponenten ohne Display

Zur Vollständigkeit wurde der Verbrauch der übrigen Systemkomponenten ohne Display-Teil bzw. Displaytreiber untersucht. Das System wurde dabei wie oben durch die SMU versorgt, der Displayteil wurde extern durch ein Labornetzteil versorgt. Die Messperiode der SMU betrug dabei 200 µs.

Die Leistungsaufnahme der übrigen Systemkomponenten ist im Schlafmodus zwischen $2\,s < t > 17\,s$ um Größenordnungen geringer als im Aktivmodus. Hier kumuliert sich der Verbrauch über 15 s auf etwa 260 mWs, womit eine durchschnittliche Leistungsaufnahme von $17,3\,mW$ folgt, was um den Faktor 17 kleiner ist, als der Verbrauch des Displays bzw. des Displaytreibers.

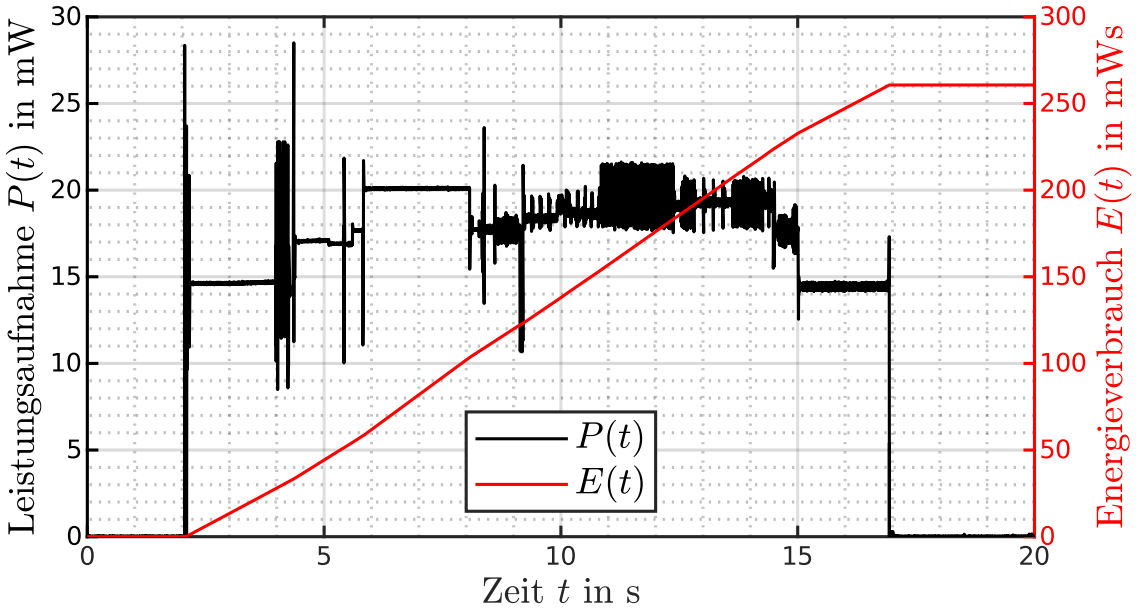


Abbildung 5.4.3: Messergebnisse zum Energieverbrauch der Systemkomponenten ohne Display.

5.5 Anzahl möglicher Updates

Aus dem Energieverbrauch pro Update und der in Abhängigkeit von der Beleuchtungsstärke verfügbaren Leistung wurde die Anzahl der im Mittel pro Tag möglichen Updates berechnet. Abb. 5.5.1 zeigt die Messergebnisse.

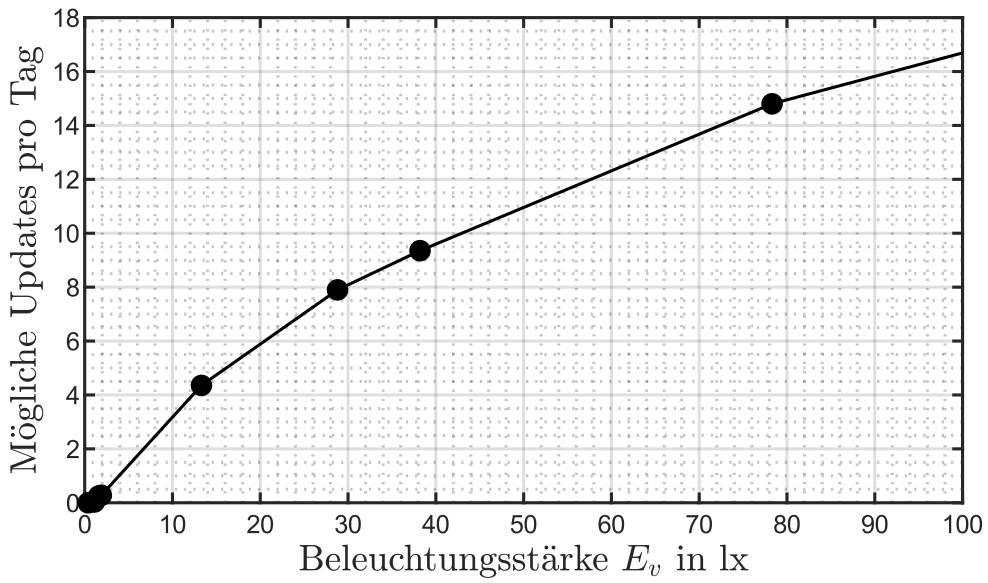


Abbildung 5.5.1: Messergebnisse zum Energieverbrauch der Systemkomponenten ohne Display.

Es zeigt sich, dass bereits bei einer Beleuchtung von 10 lx im Mittel 3 Updates pro Tag möglich sind. Die Anzal der Updates steigt wie die durch die Solarzellen gelieferte Leistung linear mit der Beleuchtungsstärke E_v an, sodass bei 20 lx bereits 6 Updates pro Tag möglich ist. Hierbei ist zu bemerken, dass am Wochenende vermutlich keine Updates getätigt werden, sodass die unter den Werktagen verfügbare Anzahl an Updates noch etwas größer ist.

5.6 Zeitliches Systemverhalten

Hier soll das zeitliche Systemverhalten analysiert werden. Dazu wurden ein unbelegter GPIO-Pin der MCU unmittelbar bei Betreten der `main()`-Funktion auf High und erst unmittelbar vor Betreten des Tiefschlafmodus wieder auf Low gesetzt.

Ein weiterer Pin wurde für die Dauer der System-Initialisierungsphase High gesetzt.

Beide Signale wurden oszillographiert und sind in Abb. 5.6.1 als grünes `AWAKE`-Signal bzw. als `INIT`-Signal in blau dargestellt.

Zusätzlich wurde mit dem gelben `INT`-Signal die Spannung am NFC-Interrupt-Pin aufgezeichnet, das `EN_EPD`-Signal in magenta zeigt die Spannung am Enable-Pin des Load-Switches zum aktiv-Schalten der Display-Spannungsversorgung. Die horizontale Zeitauflösung liegt bei 2 s/`DIV`.

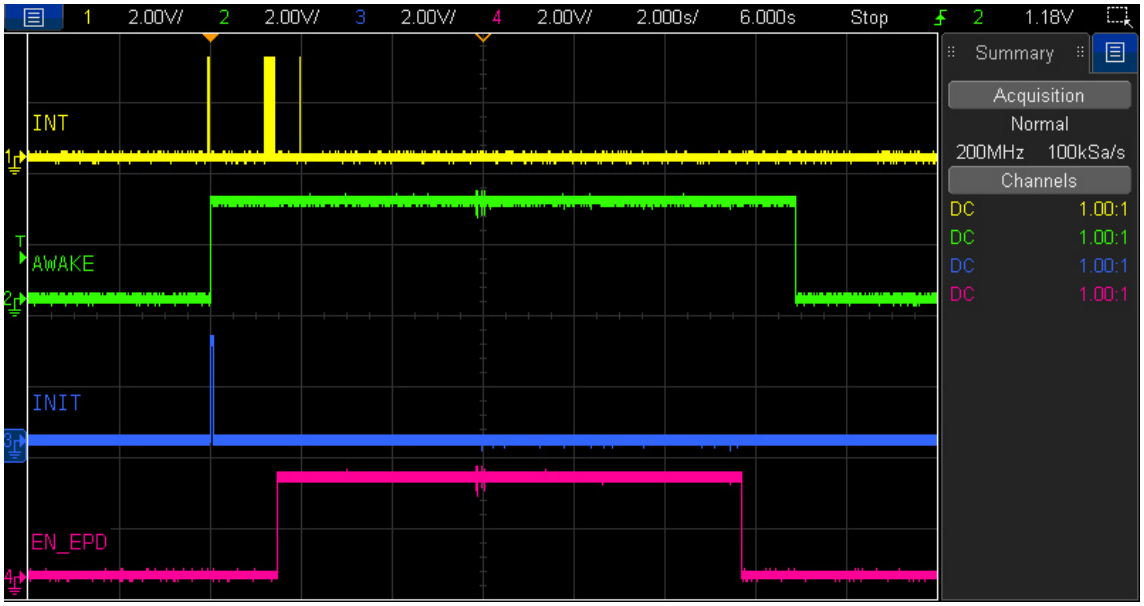


Abbildung 5.6.1: Timingverhalten des Systems während eines Updates.

Es zeigt sich, dass das System nach dem ersten Interrupt-Signal, verursacht durch das nun am NFC-Transceiver erstmals anliegenden NFC-Feld, sehr schnell in den

Aktivmodus wechselt und sich initialisiert. Die nachfolgende Totzeit von etwa einer Sekunde schwankt leicht von Messung zu Messung, die Ursache für diese konnte noch nicht ausfindig gemacht werden. Möglicherweise ist diese auch nicht wesentlich zu minimieren, sondern wird von App und NFC-Transceiver zum Verbindungsaufbau benötigt.

Die nächsten Interrupt-Signale und damit NFC-Nachrichten treffen dann burstartig in rascher Folge ein. Unmittelbar danach wird die Display-Versorgung zum Update eingeschaltet, es folgt mit etwas Verzögerung ein letzter Interrupt, verursacht durch das Entfernen des NFC-Feldes vom Transceiver. Das Display-Update beansprucht etwa 10 s.

Nach dem Abschalten der Display-Versorgung bleibt das System wie zu erwarten noch etwa 1 s wach und begibt sich dann wieder in den Tiefschlafmodus.

5.7 SPI-Kommunikation

Da das Displayupdate mit 10 s relativ lange dauert, wurde die Auslastung des SPI-Busses zum Display analysiert. Dazu wurden das `EN`-Signal des Display-Load-Switches sowie das `MOSI`-Signal des SPI-Busses während eines Updates oszillographiert, zusätzlich mit dem `HRDY`-Signal des Display-Controllers. Abb. 5.7.1 zeigt die Ergebnisse.

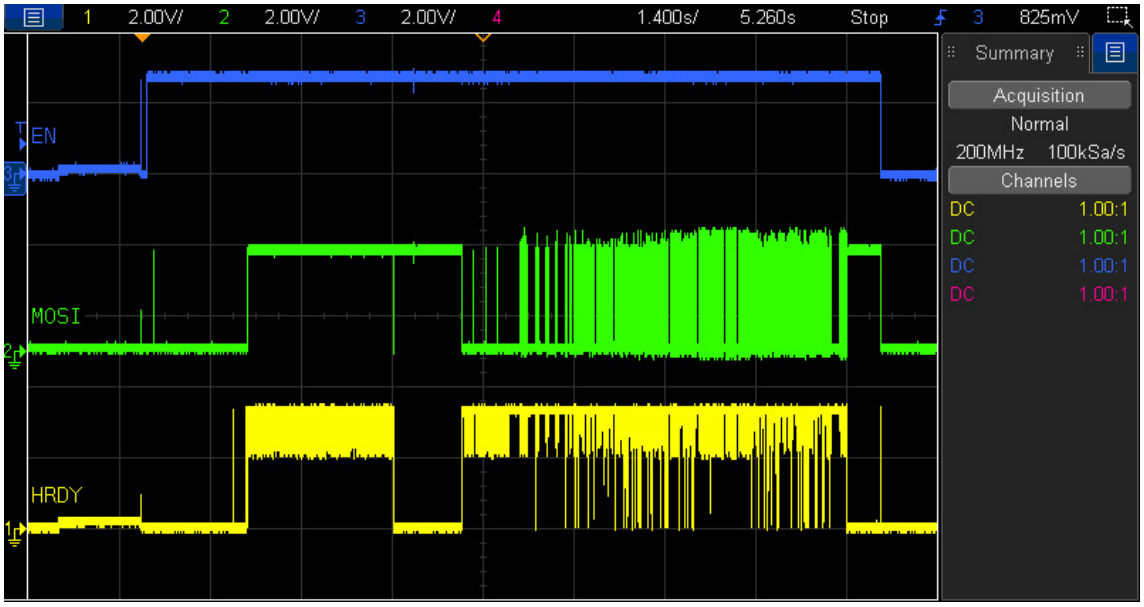


Abbildung 5.7.1: Timingverhalten des SPI-Busses.

Nach Einschalten der Spannungsversorgung, d.h. nach der steigenden Flanke des `EN`-Signals, benötigt der Display-Controller einige Zeit zur Initialisierung, erst da-

nach ist das HRDY-Signal wieder High. Danach wird das Display weiß überschrieben, erkennbar am MOSI-Signal, das dauerhaft High ist. Es folgt der erste Refresh, hier ist das HRDY-Signal wieder low. Danach wird das Türschild-Bild in den Controller geschrieben, was signifikant mehr Zeit beansprucht als das vorherige weiß-Überschreiben. Abb. 5.7.2 zeigt eine Detailaufnahme des letztgenannten Schreibprozesses.

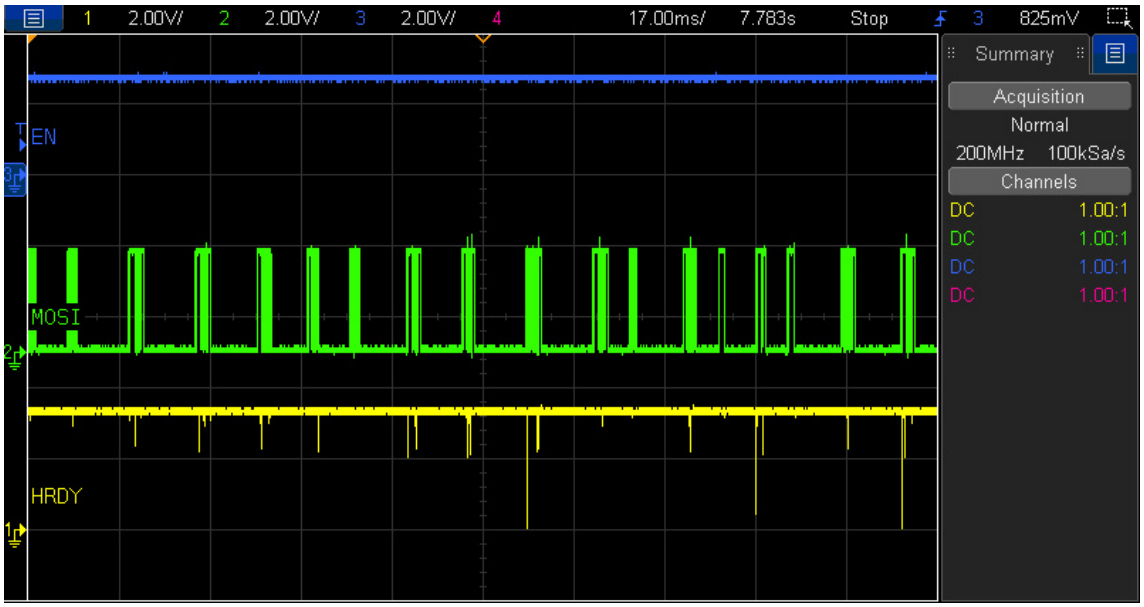
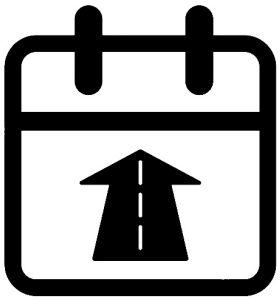


Abbildung 5.7.2: Timingverhalten des SPI-Busses, Detailaufnahme.

Die einzelnen Bursts auf dem MOSI-Signal indizieren, dass der SPI-Bus mit weniger als 50% ausgelastet ist. Dies ist nicht durch den Display-Controller begründet, da dessen HRDY-Signal nahezu dauerhaft High ist, sondern ist auf die im Vergleich zur Freqeunz der SPI-Takt-Signals von 12 MHz relativ langsame Taktrate der CPU zurückzuführen, die mit 48 MHz lediglich doppelt so groß ist.



Kapitel 6

Fazit

6.1 Erfüllung der Anforderungen

In Tab. 6.1 ist dargestellt, inwieweit das implementierte Funktionsmuster die Anforderungen erfüllt, gegebenenfalls mit Erläuterung.

Tabelle 6.1: Erfüllung Systemanforderungen.

Anforderung	Erfüllt	Bemerkung
Energieversorgung		
1a Energieautarkie	ja	-
1b Energiesparen	teilweise	Erhebliches Verbesserungspotenzial beim Display-Controller.
1c Updatezahl	ja	Bei schlechter Beleuchtung von 10 lx bis zu 3 Updates pro Tag.
1d Aufhängungsort	ja	Betrieb auch bei schlechtester Beleuchtung von 5 lx möglich.
Display		
2a E-Paper-Display	ja	-
2b Displaygröße	ja	-
2c Abwensendheits-Nachrichten	ja	Displayupdate dauert mit 10 s etwas lange.
Benutzerschnittstelle		
3a Smartphone-Kofiguration	ja	-
3b NFC-Schnittstelle	ja	-
3c Sichere-Schnittstelle	nein	Pin-Funktion vorhanden, keine Änderungs-funktion, keine Verschlüsselung.
Smartphone-App		
4a Smartphone-Anforderungen	ja	-

Hieraus ergeben sich unmittelbar Verbesserungspotenziale, die im nächsten Abschnitt erläutert werden. Insgesamt zeigt sich aber, dass das Funktionsmuser die Anforderungen gut erfüllt. Trotz der sehr hohen Leistungsaufnahme des Display-

Controllers während nahezu der gesamten Update-Zeit, werden unter schlechter Beleuchtung von 10 lx bis zu 3 Updates pro Tag ermöglicht, was letztlich durch die äußerst niedrigen Leistungsaufnahme des Systems im Schlafmodus, sowie die große Fläche effizienter Solarzellen ermöglicht wird. Bei der noch zu implementierenden Funktion zur PIN-Änderung, sowie die zugehörige verschlüsslte Übertragung derselben sind aus systemtechnischer Sicht keine nennenswerten Schwierigkeiten zu erwarten, da sowohl Smartphones, als auch die MCU Koprozessoren z.B. zur AES-Verschlüsselung bereitstellen.

6.2 Optimierungspotenziale

Display-Controller Erhebliches Einsparpotenzial bietet der verwendete Display-Controller. Die Leistungsaufnahme von 300 mW auch während der SPI-Kommunikation ist kaum zu rechtfertigen, sodass es wünschenswert wäre, einen für Low-Power-Designs passenderen Controller zu finden. Im Rahmen dieser Arbeit war es nicht möglich andere Controller-ICs in Betracht zu ziehen, da das Aufzeigen der grundsätzlichen Machbarkeit und damit die Implementierung eines Funktionsmusters als Gesamtsystem im Vordergrund stand. Eine Liste möglicher Controller findet sich unter [49] bzw. [50]. Vor allem die dort genannten Controller der Firma Epson sind interessant, da diese auch in Low-Power-Handheld-Produkten wie dem Amazon Kindle 2 verwendet werden.

Der `S1D13521B` von Epson[51] beispielsweise unterstützt Displays bis zu 2048 px× 1536 px bei bis zu 5 b-Graustufen. Der Ruhestrom von $1,4\,\mu A$ bei $2,5\,V$ Spannungsversorgung lässt auf eine erheblich geringere Leistungsaufnahme schließen, als die des verwendeten `IT8951`. Zusätzlich müsste ein EPD-Power-Management-IC, und ein externer Flash vorhanden sein, weiterhin sind womöglich Treiber implementieren, was insgesamt einen beachtlichen Entwicklungsaufwand erfordert, der sicherlich mit dem dieser Arbeit zu vergleichen ist.

MCU Aufgrund der relativ langen Update-Dauer des Displays von etwa 10 s wäre es wünschenswert, einen schnelleren Mikrocontroller zu verwenden. Wie im Absch. 3.4.3 erläutert, hat der alternativ in Betracht zu ziehende `MSP432E401Y` mit 20 µA eine um Größenordnungen höheren Energieverbrauch im Tiefschlaf, was sich aber durch eine bistabile Schaltung, durch die sich der Mikrocontroller selbst von der Energieversorgung trennen kann, und die dessen Versorgung erst wieder mit einem NFC-Interrupt aktiviert, umgehen ließe. Der `MSP432E401Y` bietet mit bis zu 120 MHz Taktrate wesentlich mehr Rechenleistung.

Komprimierung der Bitmap-Daten Der enorme Speicherbedarf der Bitmap-Daten wie Schriftarten und Piktogramme ließe sich durch eine geeignete Komprimierung vermutlich wesentlich verringern. Da die Art der Daten dergestalt ist, dass lange Folgen identischer Grautöne auftreten, würde sich hier allgemein eine Entropiecodierung und insbesondere eine Lauflängencodierung anbieten.

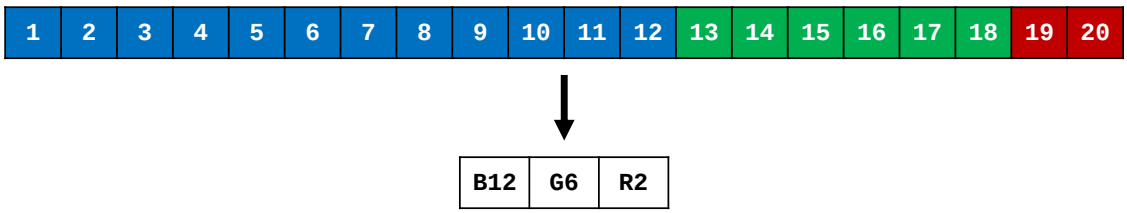


Abbildung 6.2.1: Schema einer Lauflängencodierung.

Hierbei wird, wie in Abb. 6.2.1 schematisch dargestellt, nicht die Bitmap direkt im Speicher abgebildet, sondern eine Folge von Farbwerten und zugehörigen Lauflängen, was insbesondere bei langen Widerholungen einen hohe Kompressionsrate bietet.

6.3 Zusammenfassung und Ausblick

In dieser Arbeit wurde ein Fuktionsmuster eines energieautarken Türschildes mit E-Paper-Display und NFC-Konfigurationsschnittstelle entwickelt, auf dem per Smartphone-App per NFC einfach Abwesendheitsnachrichten angezeigt werden können. Das System wird aus amorphen Silizium-Solarzellen versorgt und verfügt über einen LiPo als Energiespeicher. Durch sorgfältiges Hardware- und Software-seitiges Low-Power-Design beträgt die Leistungsaufnahme im Ruhemodus lediglich $1,5\,\mu W$. Bedingt durch den anwenderfreundlichen, jedoch für Low-Power-Designs ungeeigneten Display-Controller, beträgt der Energieverbrauch während eines Updates 300 mW. Trotzdem zeigt sich, dass das System bei einer Zellfläche von knapp $220\,cm^{-2}$ auch bei schlechter Beleuchtung in dunklen Gängen mehrere Türschild-Updates pro Tag bereitstellen kann.

Die NFC-Schnittstelle als wesentliche Neuerung zu bestehenden kommerziellen Lösungen erwies sich in der Handhabung mit dem Türschild als intuitiv, einfach und komfortabel.

Durch den Status des in dieser Arbeit entwickelten Systems als Funktionsmuster, wären als nächste Schritte zu einem Prototypen die folgenden Punkte zu bearbeiten:

1. Auswahl eines neuen, sparsameren Display-Controllers, sowie zugehöriger Peripherie und anschließende Hardware- und Software-seitige Integration.

2. Implementierung fehlender Sofwarefunktionalitäten wie PIN-Änderungsfunktion und verschlüsselte-PIN-Übertragung.

3. Entwurf eines Gehäuses.

Die Verknüpfung der drei Schlüsseltechnologien des Systems, NFC, Energy-Harvesting und E-Paper-Displays, bietet viel Potenzial, gerade im Bereich der Consumer-Elektronik. Grundsätzlich sind z.B. nicht nur Anzeigen denkbar, die sich mittels NFC konfigurieren lassen, sondern auch solche, wo über NFC weitere Informationen an ein Smartphone gesendet werden können, z.B. bei elektronischen Preisschildern. Möglicherweise kann diese Arbeit einen kleinen Teil dazu beitragen, dass der derzeit noch relativ dünne Markt für derartige Produkte weiter wächst.

Literaturverzeichnis

[1] Tiny Circuits, *Lithium Ion Polymer Battery 70mAh- ASR00011*, Eingesehen am 24.01.2020.
`https://tinycircuits.com/products/lithium-ion-polymer-battery-3-7v-70mah`.

[2] Texas Instruments, *BQ25505 - Ultra low-power boost charger with battery management and autonomous power multiplexer for primary battery in energy harvester applications*, DSLUSBJ3F. Revision E. März 2019.

[3] ST Microelectronics, *SPV1040 Datasheet - High efficiency solar battery charger with embedded MPPT*, DS6991. Revision 8. Januar 2020.

[4] ST Microelectronics, *SPV1050 - Ultralow power energy harvester and battery charger*, DocID025569. Revision 5. Mai 2018.

[5] Bohmeyer & Schuster - Schilder-Versand.com, *go2 Türschild e-Ink*, Eingesehen am 10.01.2020.
`https://www.schilder-versand.com/p/go2-tuerschild-e-ink-22011`.

[6] m.i.b Terminal, *Funk-Türschilder "door_ink"*, Eingesehen am 10.01.2020.
`https://www.mib-terminal.de/produkte/tuerschildtechnik/funk-tuerschilder-door-ink`.

[7] e-shelf-labels, *ROOMZ*, Eingesehen am 10.01.2020.
`https://www.e-shelf-labels.com/products/digital-room-signage/roomz.html`.

[8] S. Unrein, *Entwicklung eines miniaturisierten Energieversorgungs-Moduls zur autarken Versorgung von Funkmodulen*, Bachelorarbeit. Fakultät Elektrotechnik, Medizintechnik und Informatik, Hochschule Offenburg 28.02.2017.

[9] Energiebündel Weinviertel, *Wie funktioniert Photovoltaik?*, Eingesehen am 12.12.2019.

https://www.energiebuendel-weinviertel.at/wissenswertes/wie-funktioniert-photovoltaik.

[10] NREL (National Renewable Energy Laboratory), *Best Research-Cell Efficiency Chart*, Rev. 11-05-2019, May 2019.
https://www.nrel.gov/pv/cellefficiency.html.

[11] B. Minnaert and P. Veelaert, "Which type of solar cell is best for low power indoor devices?," in *i-SUP 2010 : Innovation for Sustainable Production, Proceedings*, pp. 8–12, Ghent University, Department of Electronics and information systems, 2010.

[12] A. Diehm, S. Langer, A. Angermayr u.a., *Entwicklung eines Smartphone konfigurierbaren E-Ink-Displays zur Anzeige von Informationen*, Projektarbeit im Seminar "Systementwicklung", Fakultät für Elektrotechnik, Medizintechnik, Informatik, Hochschule Offenburg. 2019.

[13] Peng Fei Bai u.a., *Review of Paper-Like Display Technologies*, Progress In Electromagnetics Research, Vol. 147, 95-115 2014.

[14] Maxwell Technologies, *3.0V 3F ULTRACAPACITOR CELL, BCAP0003 P300 X11*, Document Number: 3002552-EN.3 2019.

[15] Maxwell Technologies, *3.0V 50F ULTRACAPACITOR CELL, BCAP0050 P300 X11*, Document Number: 3002556-EN.3 2019.

[16] E. Riedel, C. Janiak, *Anorganische Chemie, 7. Aufl.* Berlin: Walter de Gruyter, 2007.

[17] K. Greschner, *Aktuelle und zukünftige Akkusysteme für die Elektromobilität. Eine vergleichende Analyse*. Hamburg: Vogel Communications Group, 2018.

[18] Lightning Global, "Lithium-ion Batteries Part I: General Overview and 2019 Update," *Technical Notes*, Juni 2019.

[19] Waveshare Wiki, *7.8inch e-Paper HAT*, Eingesehen am 24.01.2020.
https://www.waveshare.com/wiki/7.8inch_e-Paper_HAT.

[20] Waveshare Wiki, *9.7inch e-Paper HAT*, Eingesehen am 24.01.2020.
https://www.waveshare.com/wiki/9.7inch_e-Paper_HAT.

[21] ITE Tech. Inc., *IT8951 EPD Timing Controller Preliminary Specification*, V0.2.4.3 2017.

[22] Panasonic Industry, *Amorphous Silicon Solar Cells Amourphous Photosensors*, Revision April 2019.

[23] Texas Instruments, *User's Guide - MSP430FR5994 LaunchPad Development Kit (MSP430-EXP430FR5994)*, SLAU678B. September 2019.

[24] Texas Instruments, *MSP430FR599x, MSP430FR596x Mixed-Signal Microcontrollers*, SLASE54C, Revision August 2018. März 2016.

[25] Texas Instruments, *MSP432P411x, MSP432P401x SimpleLink Mixed-Signal Microcontrollers*, SLASEA0B, Revision Juni 2019. Dezember 2017.

[26] Texas Instruments, *MSP432E401Y SimpleLink Ethernet Microcontrollers*, SLASEN5. Oktober 2017.

[27] ARM - ARM Info Center, *Cortex-M Debug Connectors*, Eingesehen am 28.01.2019. `http://infocenter.arm.com/help/topic/com.arm.doc.faqs/attached/13634/cortex_debug_connectors.pdf`.

[28] Texas Instruments, *SimpleLink Ethernet MSP432E401Y Microcontroller LauchPad Development Kit (MSP-EXP432E401Y)*, SLAU748B, Revision September 2018. Oktober 2017.

[29] Texas Instruments, *MSP-EXP432E401Y Schematic*, MCU02, Revision 1.0. 12. Oktober 2017.

[30] Waveshare Wiki, *e-Paper IT8951 Driver HAT (B) Schematic*, Eingesehen am 29.01.2019. 21. September 2018. `https://www.waveshare.com/wiki/File:E-Paper-IT8951-Driver-HAT-Schematic.pdf`.

[31] Texas Instruments, *TPS65185x PMIC for E Ink Vizplex Enabled Electronic Paper Display*, SLVSAQ8G, Revision September 2017. Februar 2011.

[32] Texas Instruments, *TPS6123x 8-A Valley Current Synchronous Boost Converters with Constant Current Output Feature*, SLVSCK4A, Revision Mai 2016. September 2015.

[33] Texas Instruments, *TPS6305x Single Inductor Buck-Boost With 1-A Switches and Adjustable Soft Start*, SLVSAM8D, Revision August 2019. Juli 2013.

[34] Würth Elektronik, *WE-LQS SMT Power Inductor 74404084015*, Revision 001.005. 21. März 2019.

[35] G. Hagmann, *Grundlagen der Elektrotechnik*. AULA-Verlag, 16. Aufl. ed., 2013.

[36] H. Frohne, K. Löcherer, H. Müller, *Moeller Grundlagen der Elektrotechnik*. 19. aufl. ed.

[37] T. Grießhammer, "Entwicklung einer NFC-Antennenanpassung mit kopplungsabhängiger Leistungsregelung," Bachelorarbeit. 23. Mai 2017.

[38] J. Langer, M. Roland, *Anwendungen und Technik von Near Field Communication (NFC)*. Springer-Verlag, 2010.

[39] Texas Instruments, *HF Antenna Design Notes - Technical Appplication Report: Radio Frequency Identification Systems*, SCBA034, Revision 11-08-26-003. September 2003.

[40] ST Microelectronics, *Application note - How to design a 13.56 MHz customized antenna for ST25 NFC / RFIOD Tags*, AN2866, Revision 3. Februar 2019.

[41] ST Micreelectronics, *eDesign Antenna*, Eingesehen am 10.01.2020. `https://my.st.com/analogsimulator/html_app/antenna/#/`.

[42] P. Moser, *Hochminiaturisiertes nicht-invasives Messsystem zur Erfassung von Vitalparametern bei Kleinstlebewesen mit drahtloser RFID-/NFC-Ausleseschnittstelle*, Masterarbeit. Fakultät Elektrotechnik, Medizintechnik und Informatik, Hochschule Offenburg 31.07.2019.

[43] ST Microelectronics, *UM2306 User manual - ST25 software development kit*, DocID031172, Rev 1. November 2017.

[44] ST Micreelectronics: ST-Community, *Android SDK: readMailboxMessage() questions*, 31. Juli 2018. Eingesehen am 02.02.2020. `https://community.st.com/s/question/0D50X00009XkWPjSAN/android-sdk-readmailboxmessage-questions`.

[45] D. Gerhard, M. Schirmeister, *Entwicklung eines elektronischen Türschildes mit E-Paper-Display und Ethernet-Schnittstelle*, Projektarbeit Labor Ängewandte Forschung", Fakultät für Elektrotechnik, Medizintechnik und Informatik, Hochschule Offenburg. März 2020.

[46] SEGGER Microcontroller GmbH, *Bitmap Converter for emWin*, Eingesehen am 24.01.2020. `https://www.segger.com/products/user-interface/emwin/tools/tools-overview/`.

[47] Android Developers Reference, *Activity*, Eingesehen am 25.01.2020. `https://developer.android.com/reference/android/app/Activity`.

[48] Android Developers Reference, *Fragments*, Eingesehen am 25.01.2020. `https://developer.android.com/reference/android/app/Activity`.

[49] MobileRead Wiki, *E Ink display - E Ink controlers*, Eingesehen am 20.02.2020. `https://wiki.mobileread.com/wiki/E_Ink_display#E_Ink_controllers`.

[50] MobileRead Wiki, *Epson controller*, Eingesehen am 20.02.2020. `https://wiki.mobileread.com/wiki/Epson_controller`.

[51] Seiko Epson Corporation, *S1D13521B01 Epson / E Ink Broadsheet Hardware Functional Specification*, Rev 1.2. 2008.